Artificial faces synthesized by StyleGAN (Nvidia)

COMP547

DEEP UNSUPERVISED LEARNING

Lecture #9 – Generative Adversarial Networks Part 2

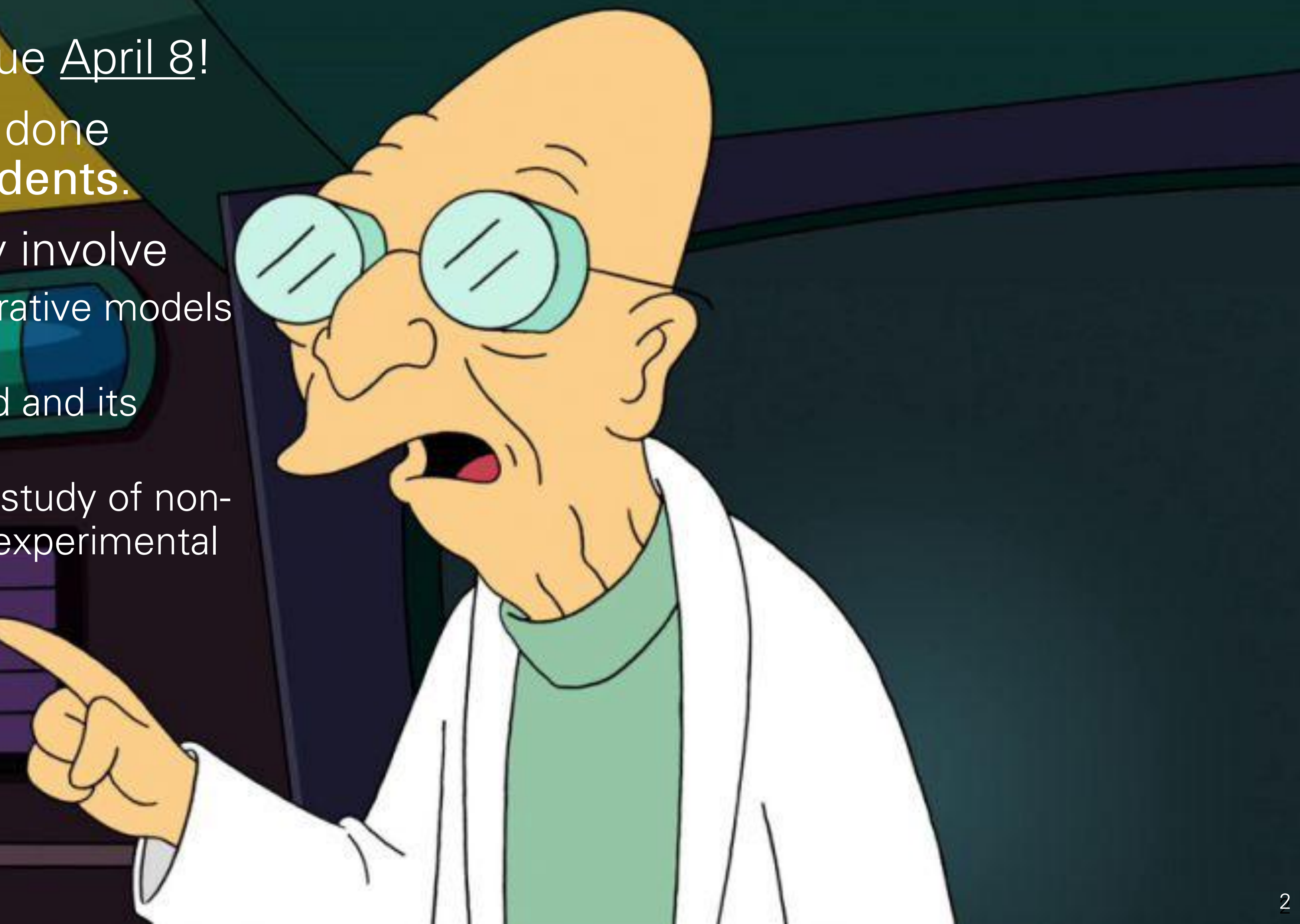
KOÇ UNIVERSITY

Aykut Erdem // Koç University // Spring 2022

Good news, everyone!

- Project proposals are due April 8!
- The projects should be done in **groups of 2 to 3 students**.
- The course project may involve
  - Application of deep generative models on a novel task/dataset.
  - Design of a novel method and its experimental analysis,
  - An extension to a recent study of non-trivial complexity and its experimental analysis.
  - Reproduction of a work published in recent years

    If you chose this particular path, participation to ML Reproducibility Challenge is strongly encouraged!



Good news, e

- Project proposals are du
- The projects should be c in **groups of 2 to 3 stud**
- The course project may
  - Application of deep genera on a novel task/dataset.
  - Design of a novel method experimental analysis,
  - An extension to a recent s trivial complexity and its ex analysis.
  - Reproduction of a work published in recent years

  If you chose this particular path, participation to ML Reproducibility Challenge is strongly encouraged!

example_proposal.pdf
1 page

Project Title

Name Surname [*1]

Abstract

1. Introduction

Introduce the task that you are going to investigate in your course project. State why you find your project topic interesting and what is difficult about it.

2. Related Work

Review previous work most relevant to your project topic. Discuss how you might improve upon these existing approaches.

3. The Approach

Give a brief outline of your approach. Describe the architecture you will use, whether you will extend an existing implementation, etc. Please note that you can change your approach later.

4. Experimental Evaluation

Explain which dataset(s) you will use to train and test your model. Describe how you will evaluate the performance of your approach against those of competing methods.

5. Work Plan

Provide a rough timeline about the planned activities and their approximate deadlines. For example,

Activity	Deadline
Complete the literature search	MM/DD/YY
Reproduce results of a baseline approach	MM/DD/YY
Prepare progress report	MM/DD/YY
Make improvements X, Y, Z	MM/DD/YY
Prepare final report and presentation	MM/DD/YY

(Hinton & Salakhutdinov, 2006; Goodfellow et al., 2014)

[*]Equal contribution [1]Department of Computer Engineering. Correspondence to: Name Surname <email>.

COMP547 Deep Unsupervised Learning, Spring 2022.

References

Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., and Bengio, Y. Generative adversarial nets. In Ghahramani, Z., Welling, M., Cortes, C., Lawrence, N., and Weinberger, K. Q. (eds.), *Advances in Neural Information Processing Systems*, volume 27, pp. 2672–2680, 2014.

Hinton, G. E. and Salakhutdinov, R. R. Reducing the dimensionality of data with neural networks. *Science*, 313: 504 – 507, 2006.

Lecture overview

- Motivation and Definition of Implicit Models
- Original GAN (Goodfellow et al, 2014)
- Evaluation: Parzen, Inception, Frechet
- Theory of GANs
- **GAN Progression**
  - DC GAN (Radford et al, 2016)
  - Improved Training of GANs (Salimans et al'16), Projected GAN (Sauer et al'21) WGAN, WGAN-GP, Progressive GAN, SN-GAN, SAGAN
  - BigGAN, BigGAN-Deep, **StyleGAN, StyleGAN2**, **StyleGAN3,** StyleGAN-XL, Self-Distilled StyleGAN, VIB-GAN, VQ-GAN
- Conditional GANs, Cycle-Consistent Adversarial Networks
- GANs and Representations
- Applications

StyleGAN

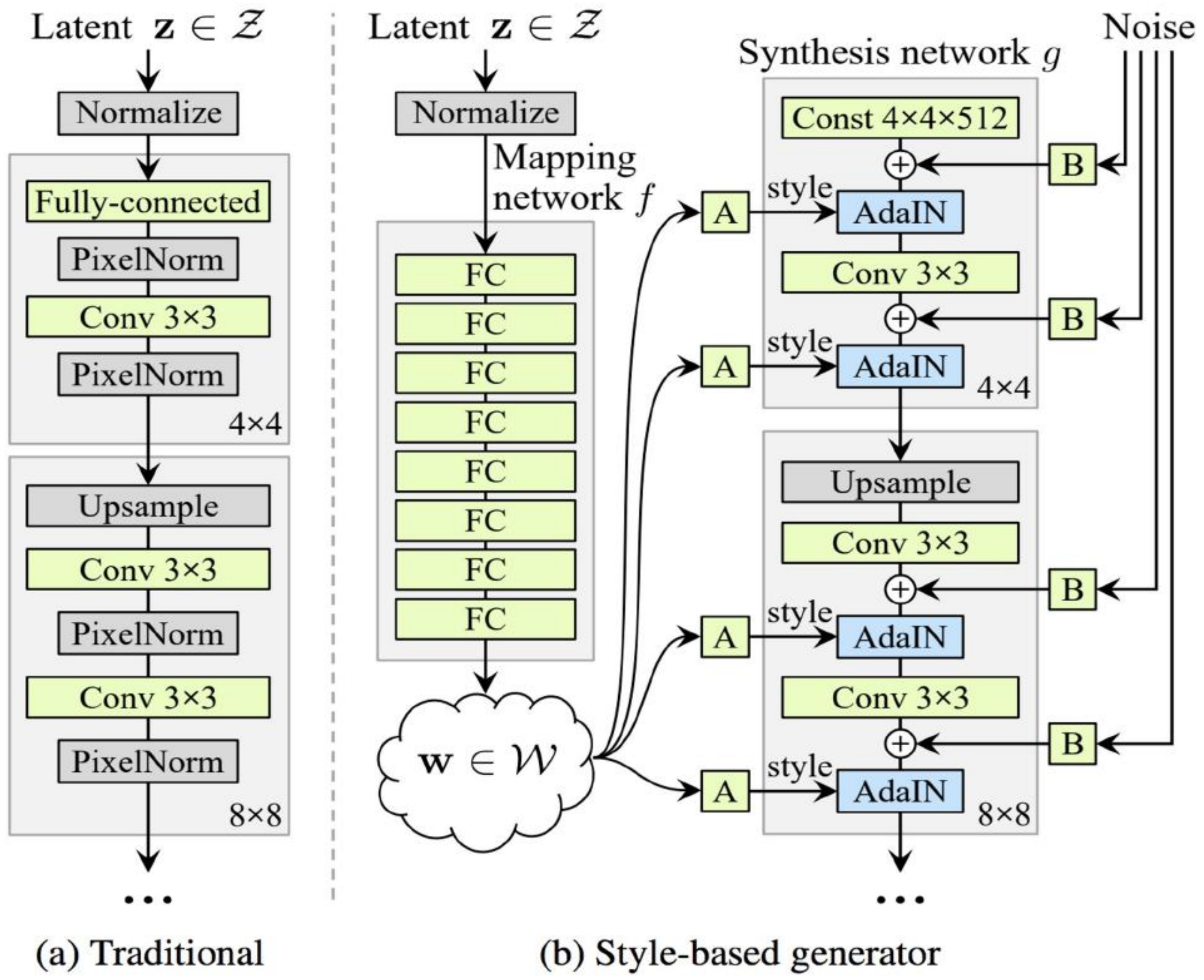
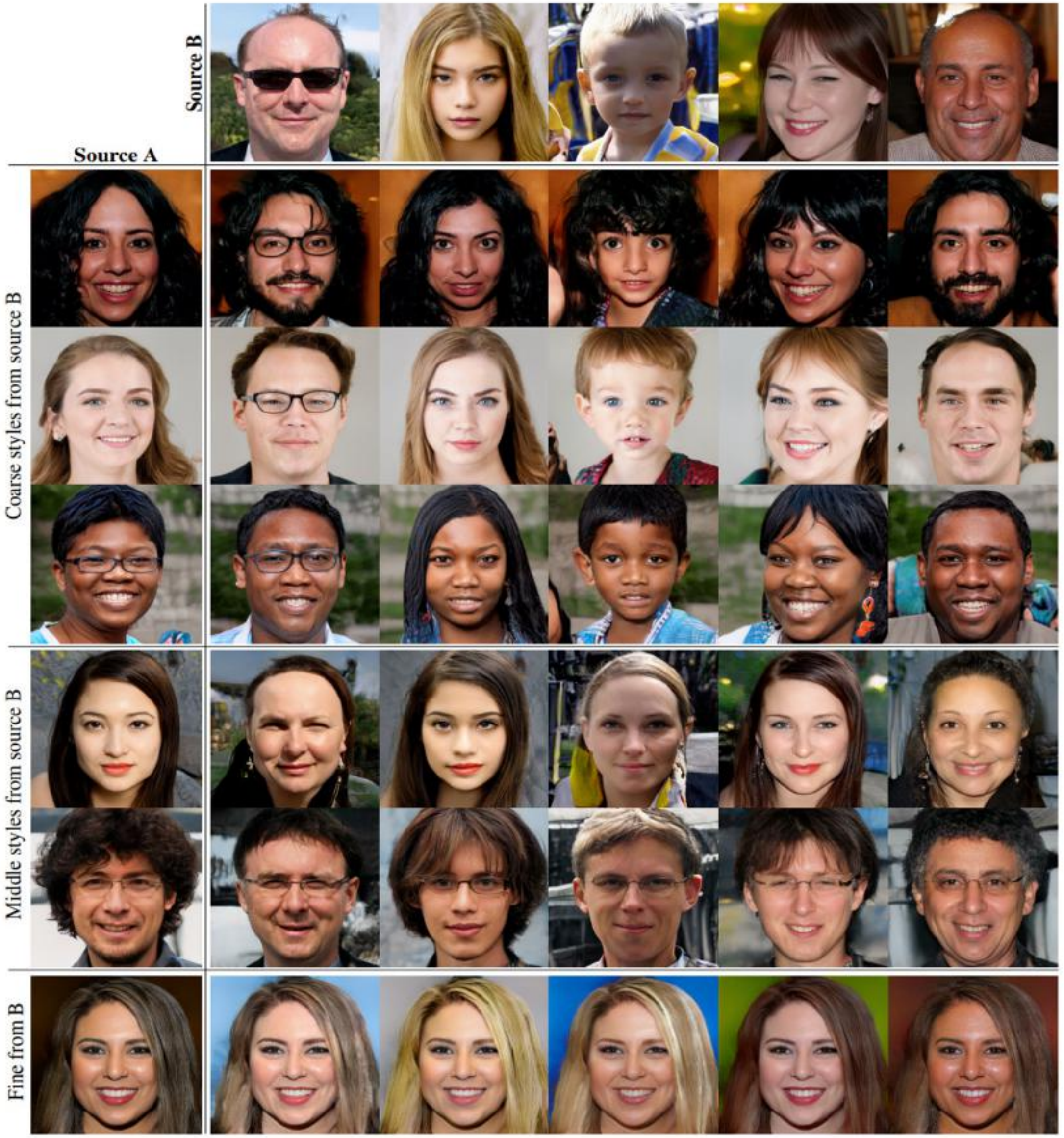


Figure 1. While a traditional generator [30] feeds the latent code though the input layer only, we first map the input to an intermediate latent space $\mathcal{W}$, which then controls the generator through adaptive instance normalization (AdaIN) at each convolution layer. Gaussian noise is added after each convolution, before evaluating the nonlinearity. Here "A" stands for a learned affine transform, and "B" applies learned per-channel scaling factors to the noise input. The mapping network f consists of 8 layers and the synthesis network g consists of 18 layers — two for each resolution ($4^2 - 1024^2$). The output of the last layer is converted to RGB using a separate 1×1 convolution, similar to Karras et al. [30]. Our generator has a total of 26.2M trainable parameters, compared to 23.1M in the traditional generator.

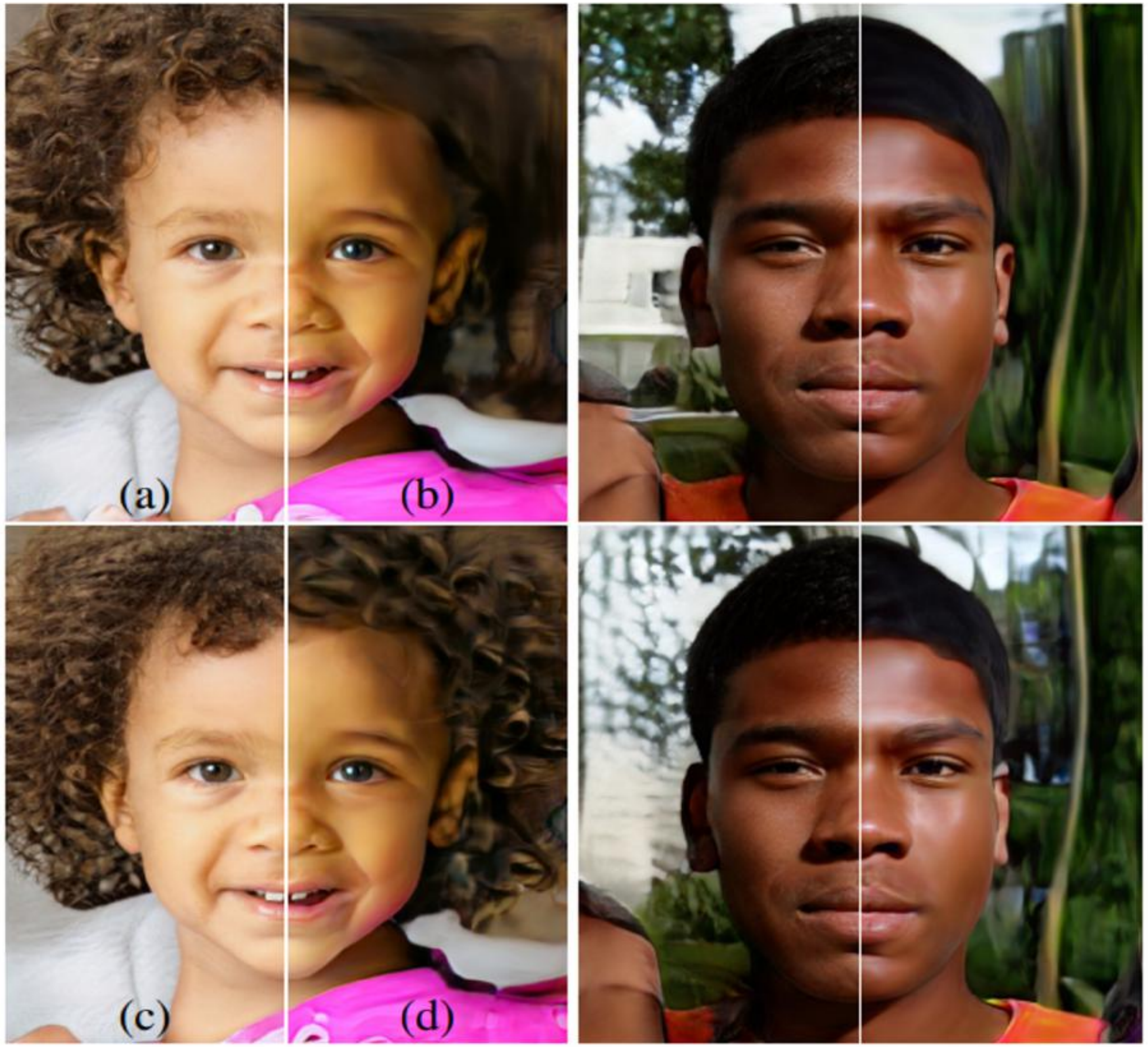
StyleGAN - Adaptive Instance Norm

$$\mathrm{AdaIN}\left(\mathbf{x}_i, \mathbf{y}\right) = \mathbf{y}_{s,i} \frac{\mathbf{x}_i - \mu\left(\mathbf{x}_i\right)}{\sigma\left(\mathbf{x}_i\right)} + \mathbf{y}_{b,i}$$

StyleGAN - Style Transfer

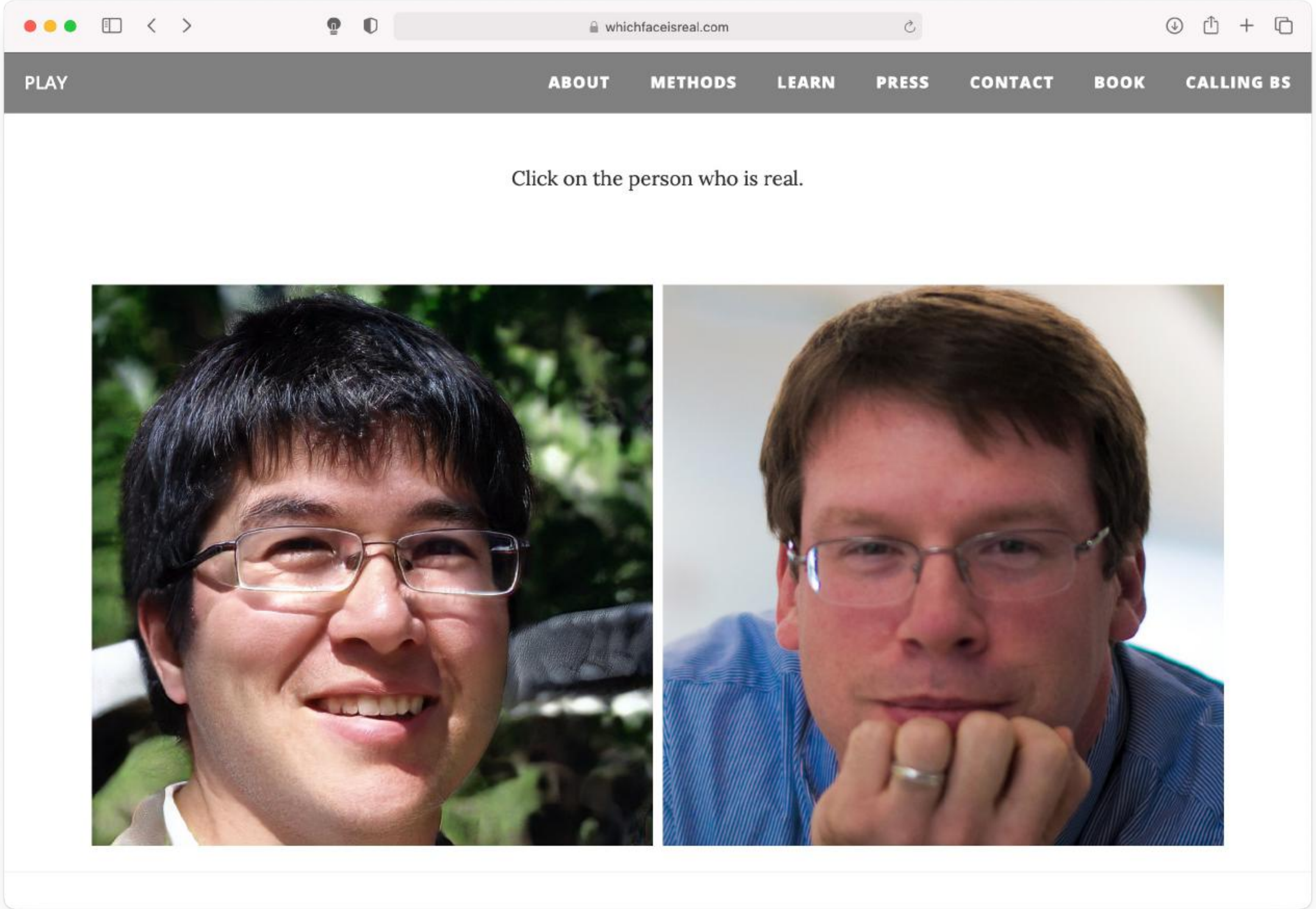


StyleGAN - Effect of adding noise



StyleGAN - Effect of noise

(a) Generated image (b) Stochastic variation (c) Standard deviation



https://www.whichfaceisreal.com/learn.html

StyleGAN Water Droplet-like Artifacts

Figure 1. Instance normalization causes water droplet -like artifacts in StyleGAN images. These are not always obvious in the generated images, but if we look at the activations inside the generator network, the problem is always there, in all feature maps starting from the 64x64 resolution. It is a systemic problem that plagues all StyleGAN images.

StyleGAN2

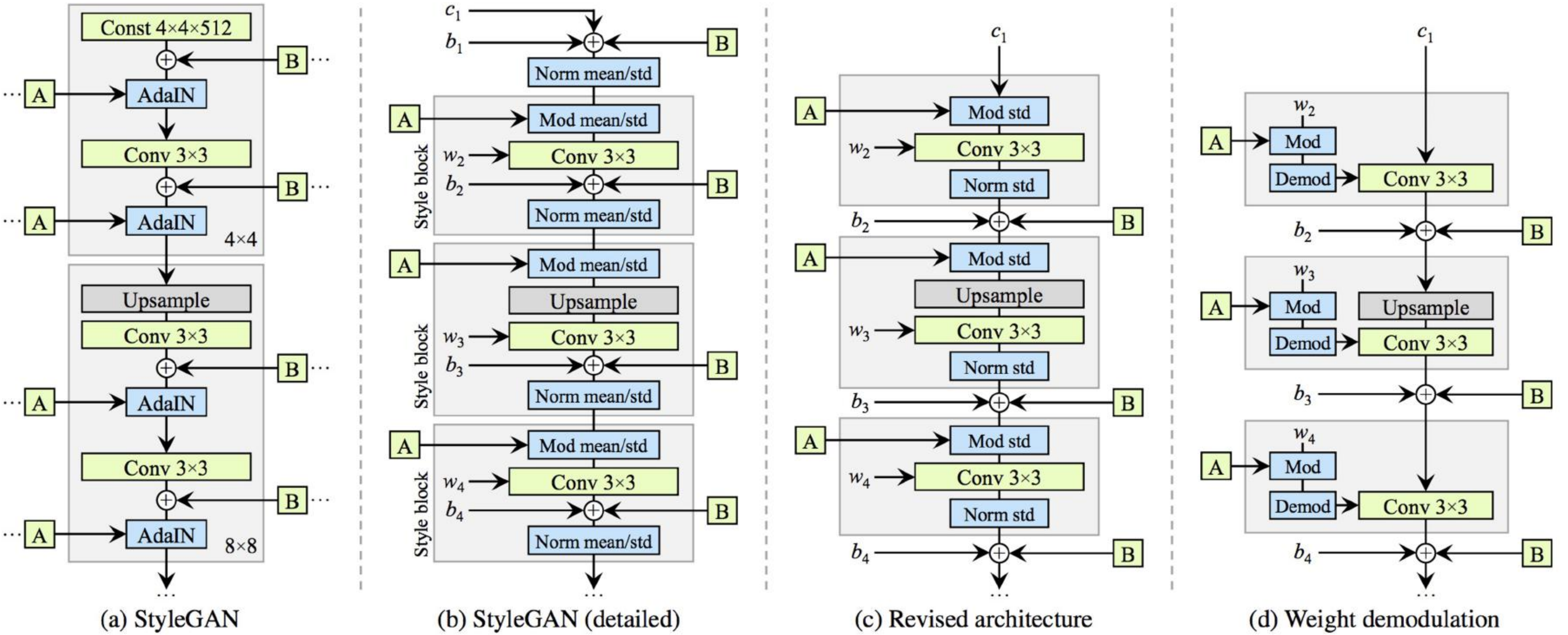


Fig. 2. We redesign the architecture of the StyleGAN synthesis network. (a) The original StyleGAN, where A denotes a learned affine transform from W that produces a style and B is a noise broadcast operation. (b) The same diagram with full detail. Here we have broken the AdaIN to explicit normalization followed by modulation, both operating on the mean and standard deviation per feature map. We have also annotated the learned weights (w), biases (b), and constant input (c), and redrawn the gray boxes so that one style is active per box. The activation function (leaky ReLU) is always applied right after adding the bias. (c) We make several changes to the original architecture that are justified in the main text. We remove some redundant operations at the beginning, move the addition of b and B to be outside active area of a style, and adjust only the standard deviation per feature map. (d) The revised architecture enables us to replace instance normalization with a "demodulation" operation, which we apply to the weights associated with each conv layer.

StyleGAN2 Phase Artifacts

Figure 6. Progressive growing leads to "phase" artifacts. In this example the teeth do not follow the pose but stay aligned to the camera, as indicated by the blue line.

StyleGAN2 Phase Artifacts

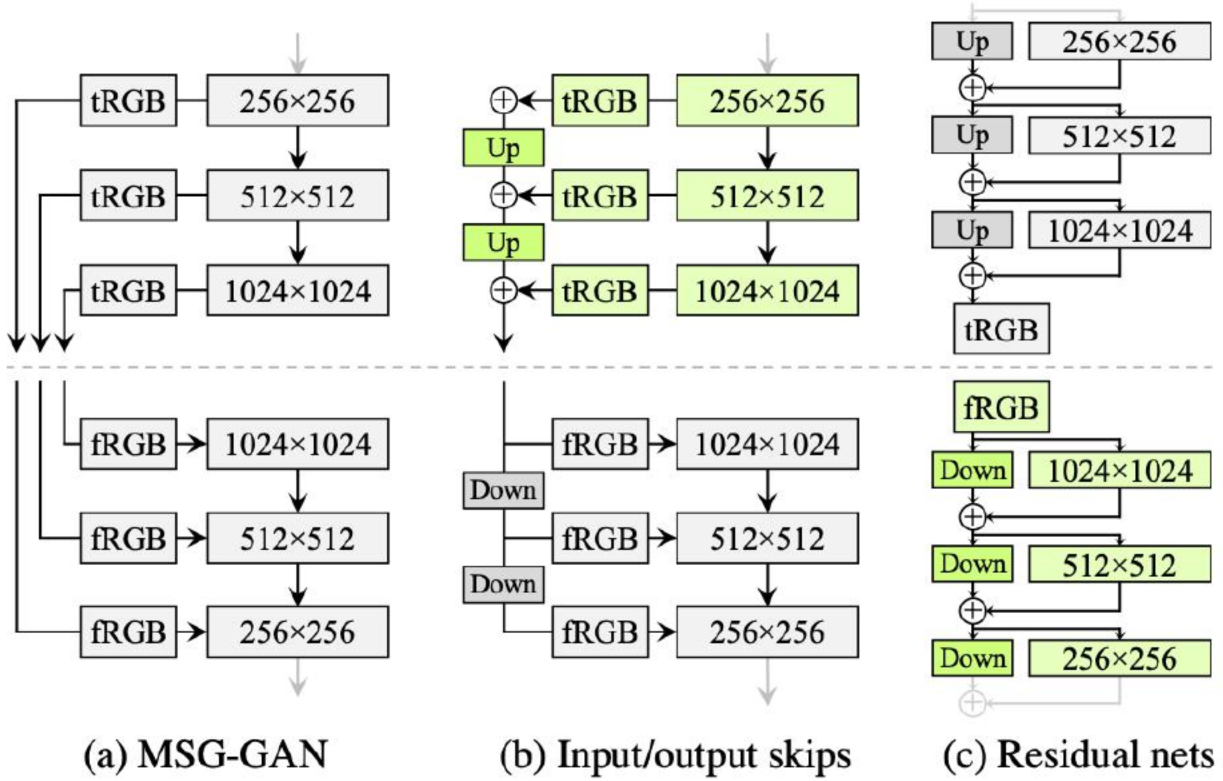
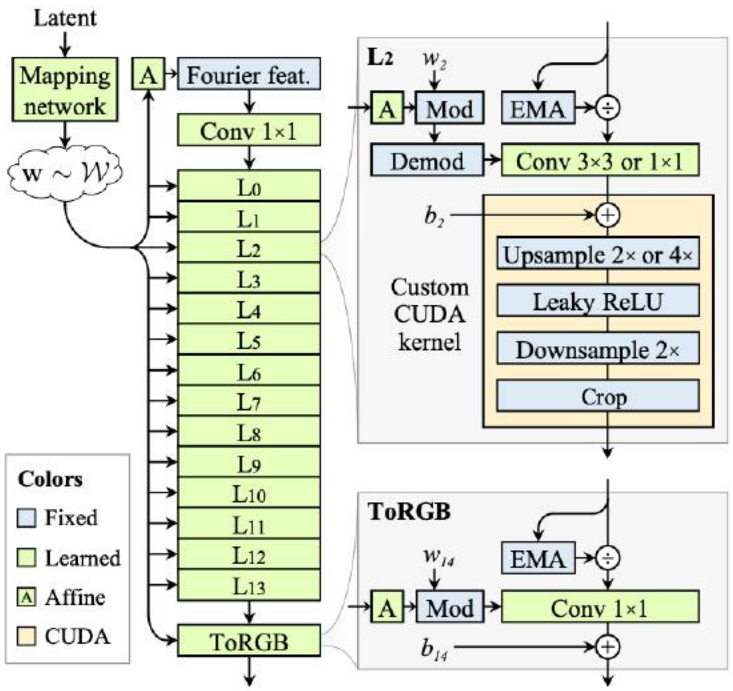


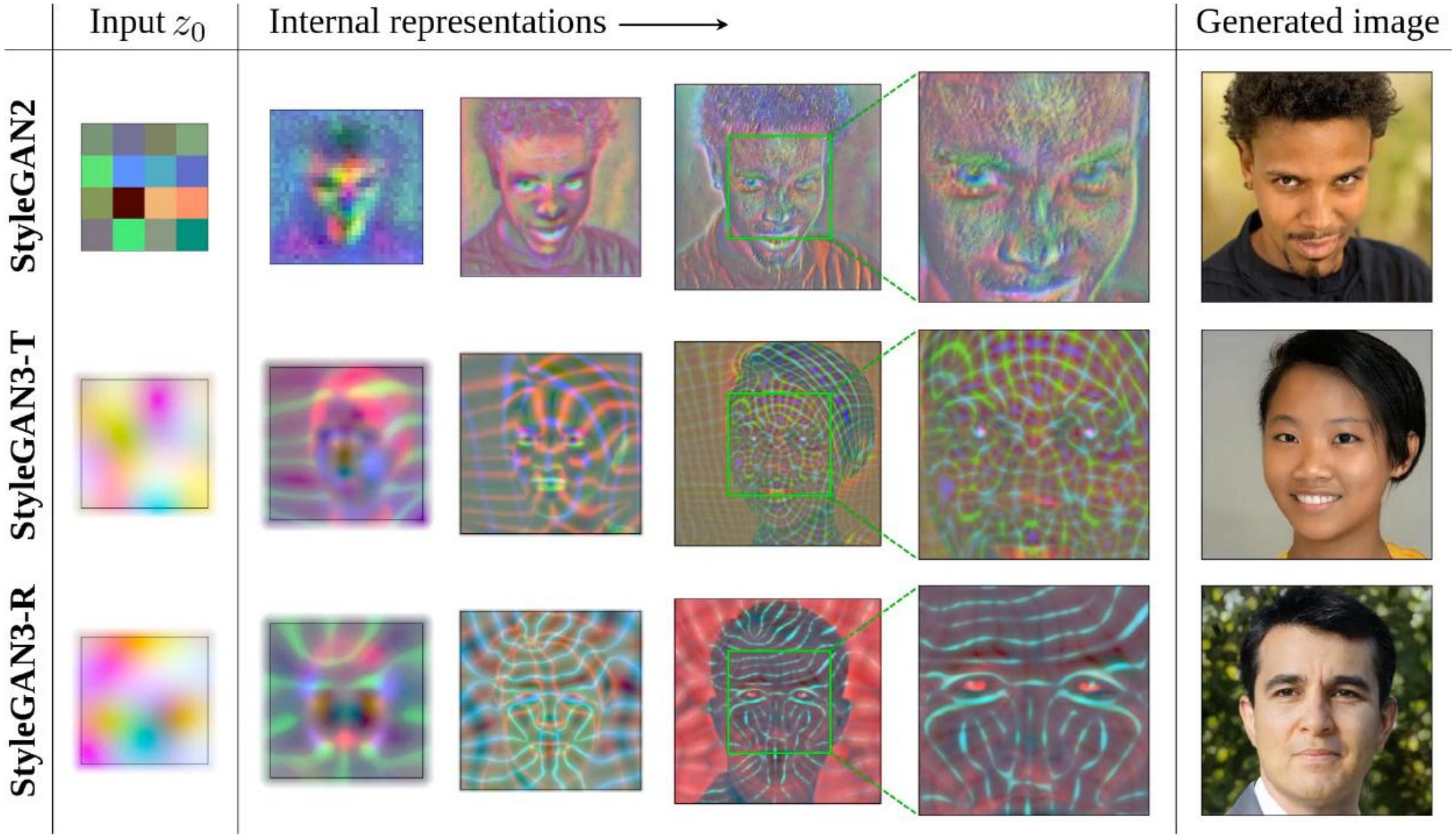
Figure 7. Three generator (above the dashed line) and discriminator architectures. Up and Down denote bilinear up and down-sampling, respectively. In residual networks these also include 1×1 convolutions to adjust the number of feature maps. tRGB and fRGB convert between RGB and high-dimensional per-pixel data. Architectures used in configs E and F are shown in green.


FFHQ, ψ = 0.50

StyleGAN3 to resolve "texture sticking"

StyleGAN3





Implementation requires custom CUDA kernel

- Internal activations encode phase information
- Fully equivariant to translation and rotation even at subpixel scale

Lecture overview

- Motivation and Definition of Implicit Models
- Original GAN (Goodfellow et al, 2014)
- Evaluation: Parzen, Inception, Frechet
- Theory of GANs
- **GAN Progression**
  - DC GAN (Radford et al, 2016)
  - Improved Training of GANs (Salimans et al'16), Projected GAN (Sauer et al'21) WGAN, WGAN-GP, Progressive GAN, SN-GAN, SAGAN
  - BigGAN, BigGAN-Deep, StyleGAN, StyleGAN2, StyleGAN3, **StyleGAN-XL**, **Self-Distilled StyleGAN**, VIB-GAN, VQ-GAN
- Conditional GANs, Cycle-Consistent Adversarial Networks
- GANs and Representations
- Applications

StyleGAN-XL

- StyleGAN was designed for controllability
- Its performance degrades on unstructured datasets such as ImageNet.

- StyleGAN-XL shows that it is possible with a carefully designed architecture and traning schemes
  - StyleGAN3 framework
  - Projected GAN objective
  - Progressive growing
  - 1024×1024 images

StyleGAN-XL

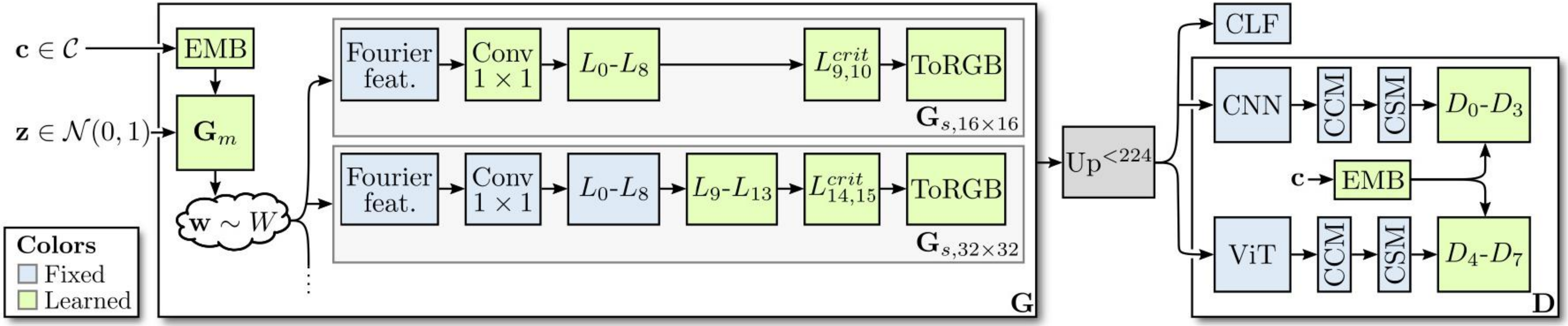


Fig. 2. **Training StyleGAN-XL**. We feed a latent code $\mathbf{z}$ and class label $\mathbf{c}$ to the pretrained embedding and the mapping network $\mathbf{G}_m$ to generate style codes $\mathbf{w}$. The codes modulate the convolutions of the synthesis network $\mathbf{G}_s$. During training, we gradually add layers to double the output resolution for each stage of the progressive growing schedule. We only train the latest layers while keeping the others fixed. The synthesized image is upsampled when smaller than 224^2 and passed through a CNN and a ViT and respective feature mixing blocks (CCM+CSM). At higher resolutions, the CNN receives the unaltered image while the ViT receives a downsampled input to keep memory requirements low but still utilize its global feedback. Finally, we apply eight independent discriminators on the resulting multi-scale feature maps. The image is also fed to classifier CLF for classifier guidance.

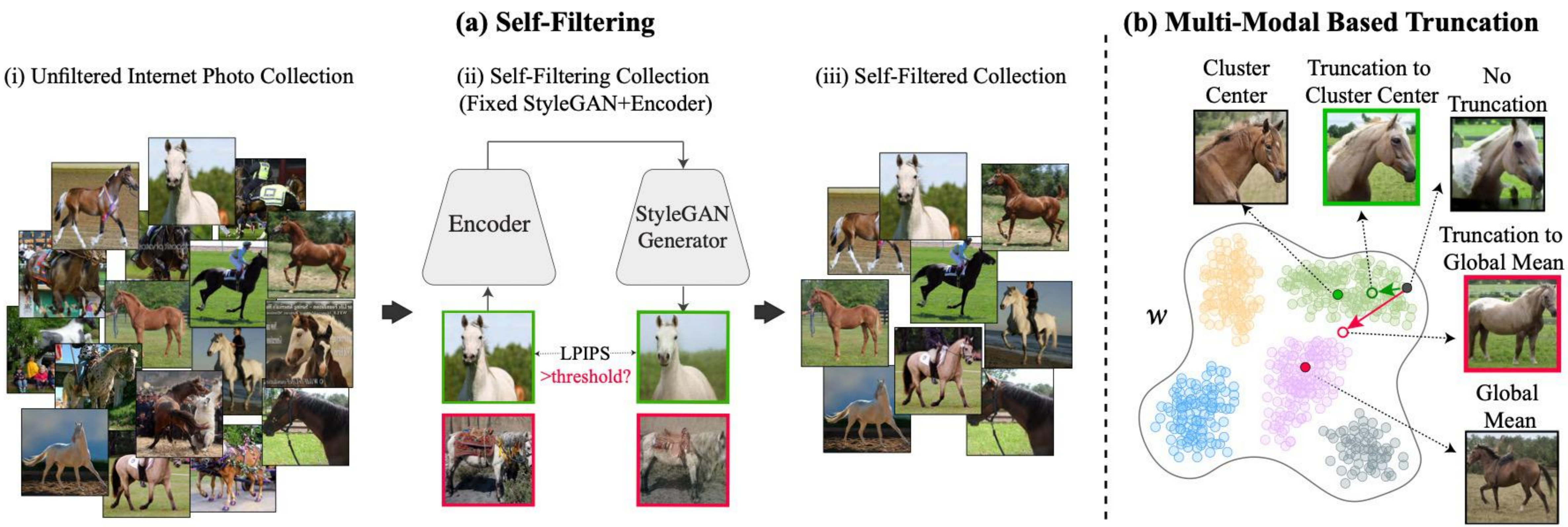
StyleGAN-XL

	Configuration	FID ↓	IS ↑
A	StyleGAN3	53.57	15.30
B	+ Projected GAN & small **z**	22.98	57.62
C	+ Pretrained embeddings	20.91	35.79
D	+ Progressive growing	19.51	35.74
E	+ ViT & CNN as $\mathbf{F}_{1,2}$	12.43	56.72
F	+ CLF guidance (StyleGAN-XL)	**12.24**	**86.21**

Fig. 3. **Samples at Different Resolutions Using the Same w.** The samples are generated by the models obtained during progressive growing. We upsample all images to 1024^2 using nearest-neighbor interpolation for visualization purposes. Zooming in is recommended.

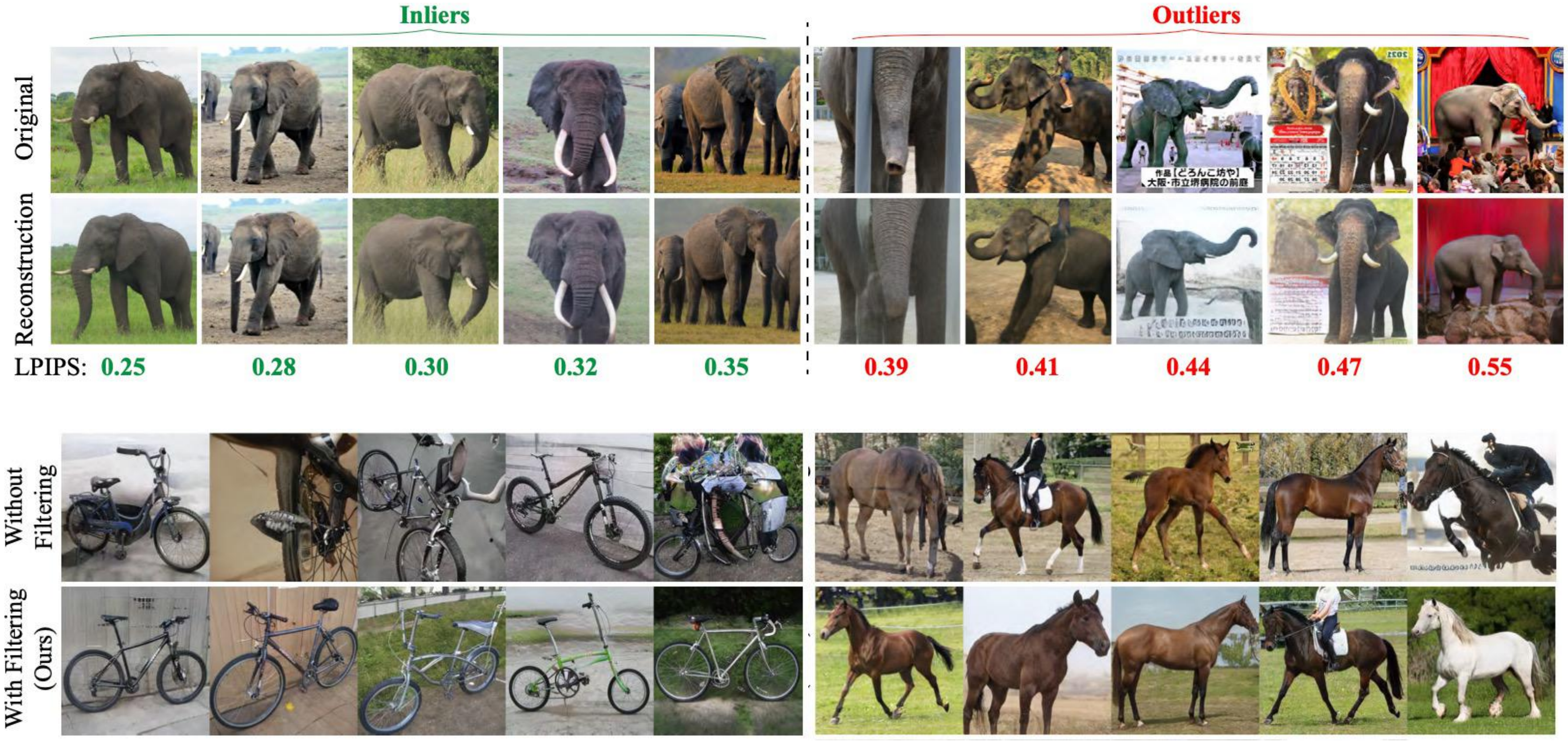
Fig. 4. **Inversion of a Given Source Image.** For BigGAN, we invert to its latent space **z**, for StyleGAN-XL we invert to style codes **w**.

Self-Distilled StyleGAN



- How to train StyleGAN on noisy Internet images?
- GAN inversion quality to automatically filter out outlier images (LPIPS)
- Multi-modal based truncation trick to cluster

Self-Distilled StyleGAN – Self-filtering



Self-Distilled StyleGAN – Multi-modal Truncation

(a) No Truncation (b) Truncation to Global Mean (c) Truncation to Cluster (Ours)

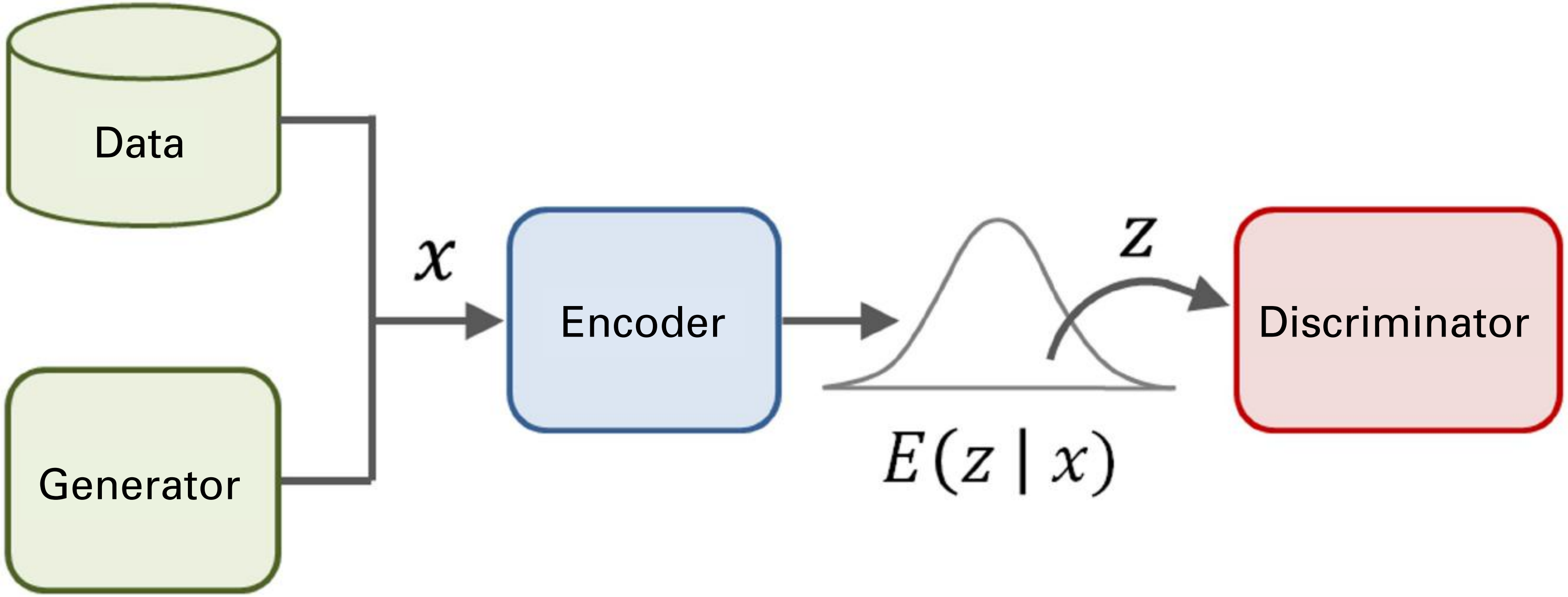
$$w_t = \psi \cdot w + (1 - \psi) \cdot c_i.$$

c_i: the “nearest” cluster center

Lecture overview

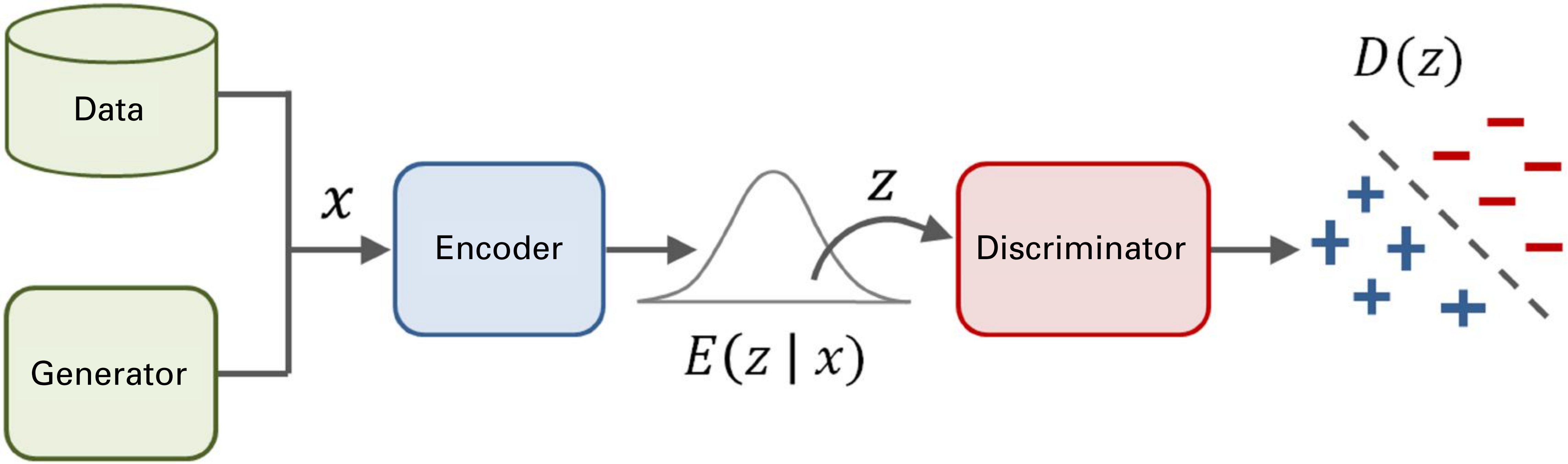
- Motivation and Definition of Implicit Models
- Original GAN (Goodfellow et al, 2014)
- Evaluation: Parzen, Inception, Frechet
- Theory of GANs
- **GAN Progression**
  - DC GAN (Radford et al, 2016)
  - Improved Training of GANs (Salimans et al'16), Projected GAN (Sauer et al'21) WGAN, WGAN-GP, Progressive GAN, SN-GAN, SAGAN
  - BigGAN, BigGAN-Deep, StyleGAN, StyleGAN2, StyleGAN3, StyleGAN-XL, Self-Distilled StyleGAN, **VIB-GAN**, VQ-GAN
- Conditional GANs
- GANs and Representations
- Applications

Information Bottleneck



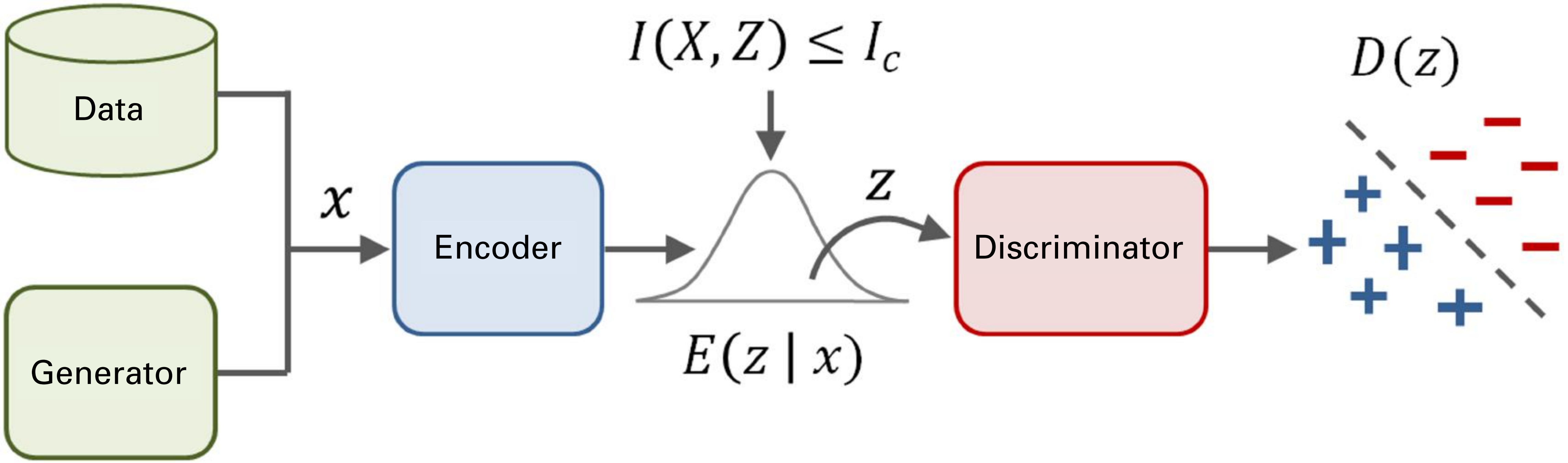
Variational Information Bottleneck [Alemi et al., 2016]
Variational Information Bottleneck GAN [Peng et al, 2019]

Information Bottleneck



Variational Information Bottleneck [Alemi et al., 2016]
Variational Information Bottleneck GAN [Peng et al, 2019]

Information Bottleneck

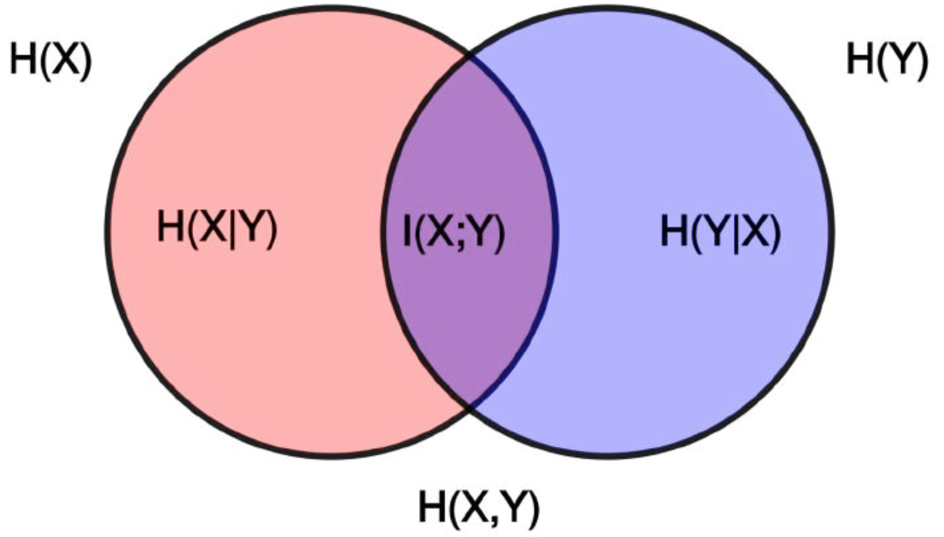


Variational Information Bottleneck [Alemi et al., 2016]
Variational Information Bottleneck GAN [Peng et al, 2019]

Mutual Information

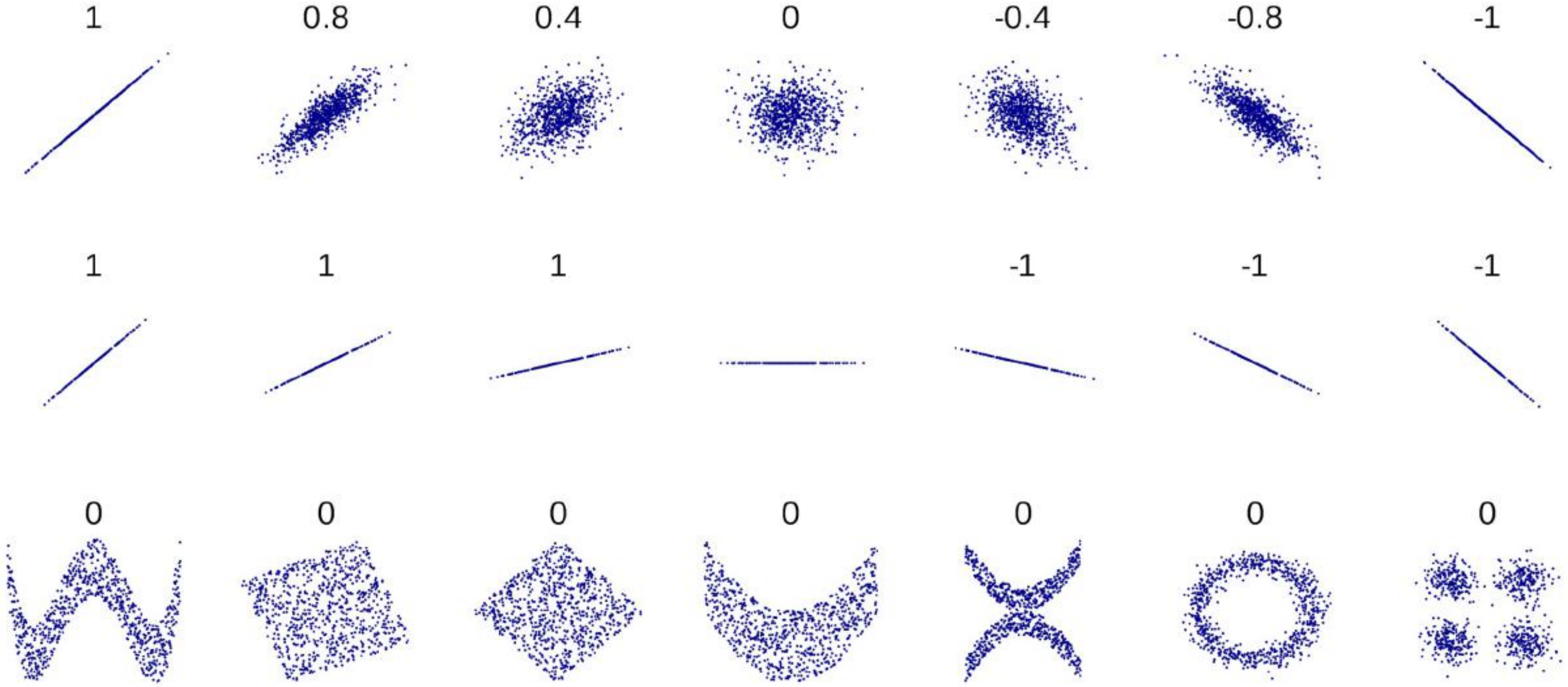
- Mutual information between two random variables X, Y: I(X; Y) is defined as

$$I(X;Y) = H(X) - H(X|Y) = H(Y) - H(Y|X)$$



Mutual Information

- Mutual Information is a general way to measure dependency between two random variables
  - Unlike the more commonly used covariance



Estimating Mutual Information

- We can try to estimate the mutual information between z and x in a latent variable model

$$
\begin{aligned}
I(z;x) &= H(z) - H(z|x) \\
&= H(z) - \mathbb{E}_{(z,x)\sim p(z,x)}[-\log p(z|x)] \\
&= H(z) + \mathbb{E}_{(z,x)\sim p(z,x)}[\log p(z|x) - \log q(z|x) + \log q(z|x)] \\
&\geq H(z) + \mathbb{E}_{(z,x)\sim p(z,x)}[\log q(z|x)]
\end{aligned}
$$

- Has intractable posterior p(z|x) but we can estimate by introducing a variational distribution q(z|x)

Information Bottleneck

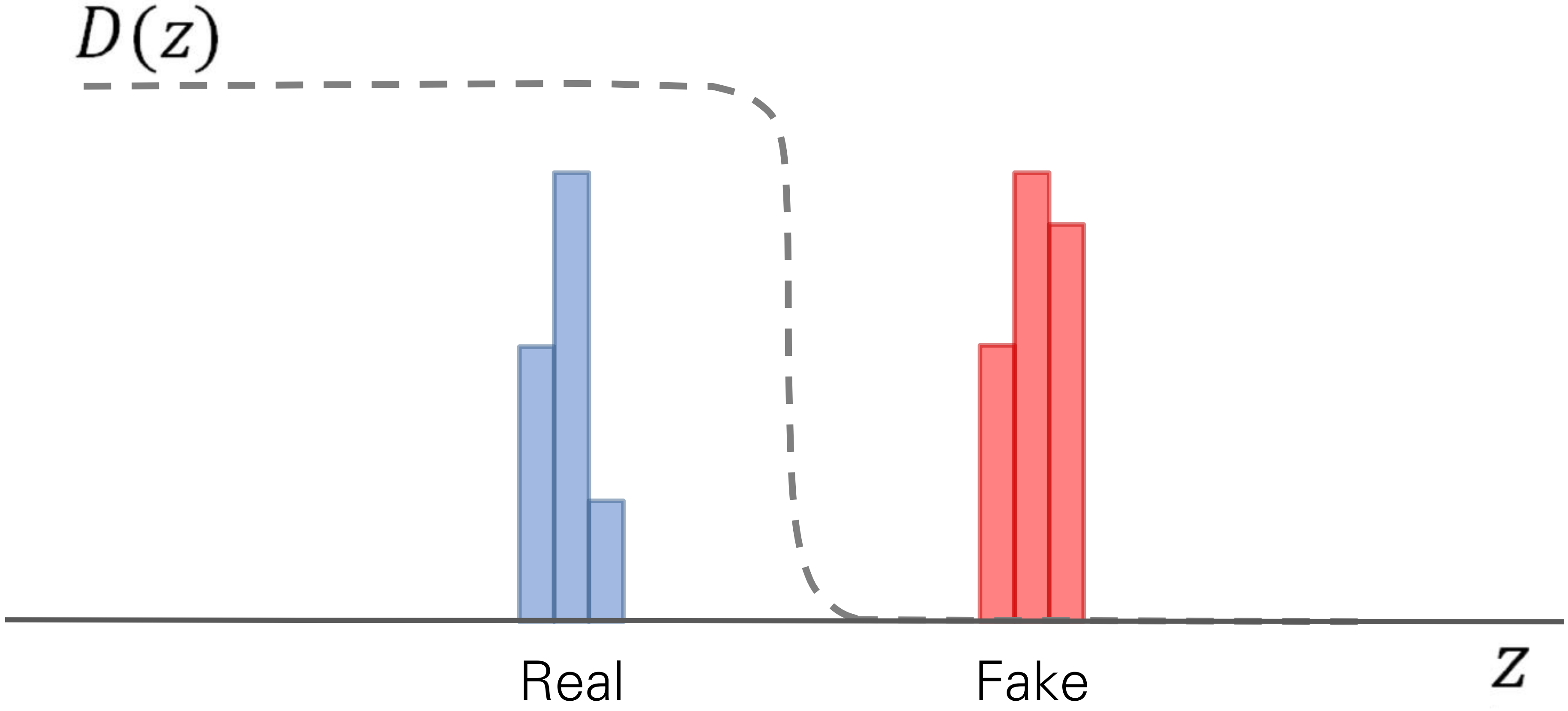
$$I(X,Z) \leq I_c$$

$$\Downarrow$$

$$\mathbb{E}_{\mathbf{x}\sim\tilde{p}(\mathbf{x})}\left[\mathrm{KL}\left[E(\mathbf{z}|\mathbf{x})\|r(\mathbf{z})\right]\right] \leq I_c$$

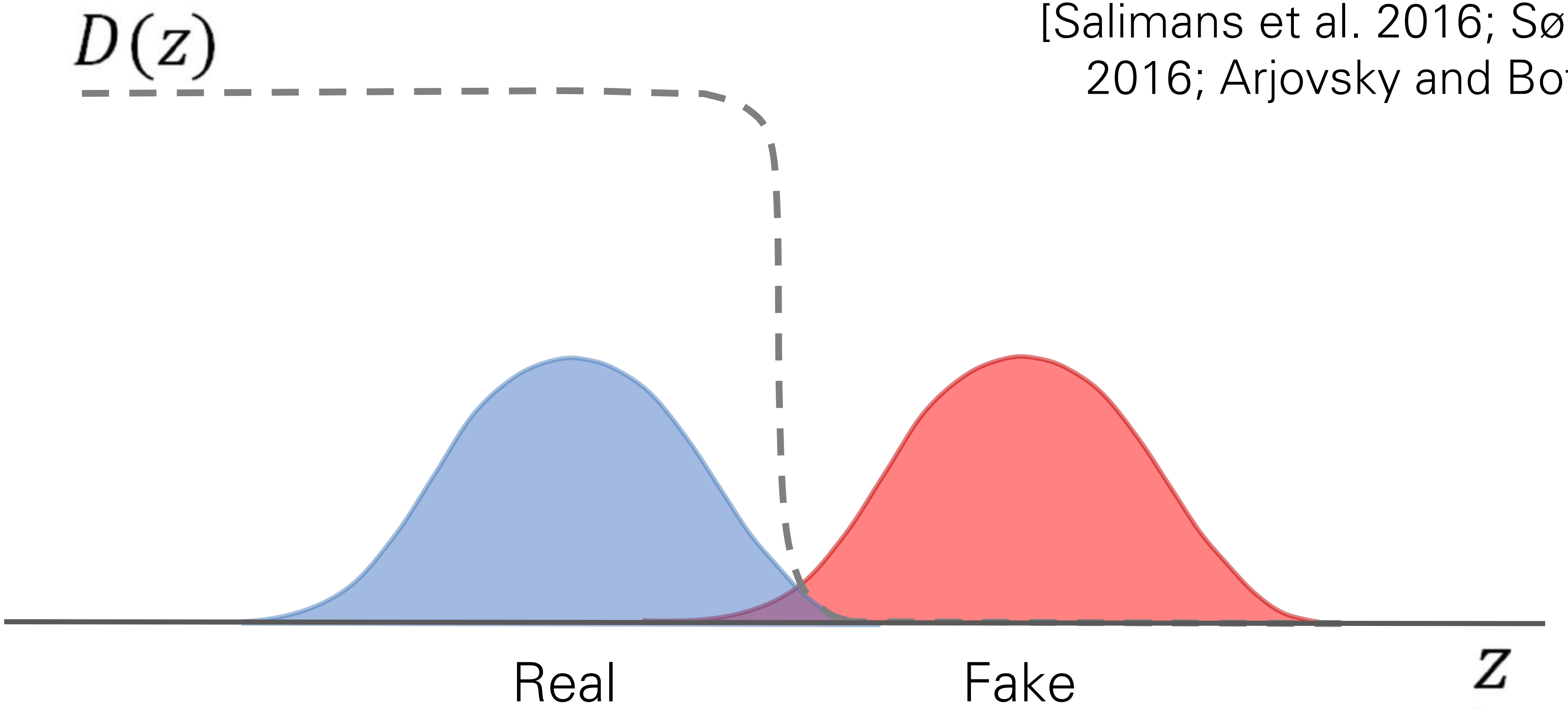
Variational Information Bottleneck (VIB)
[Alemi et al., 2016]

Variational Information Bottleneck



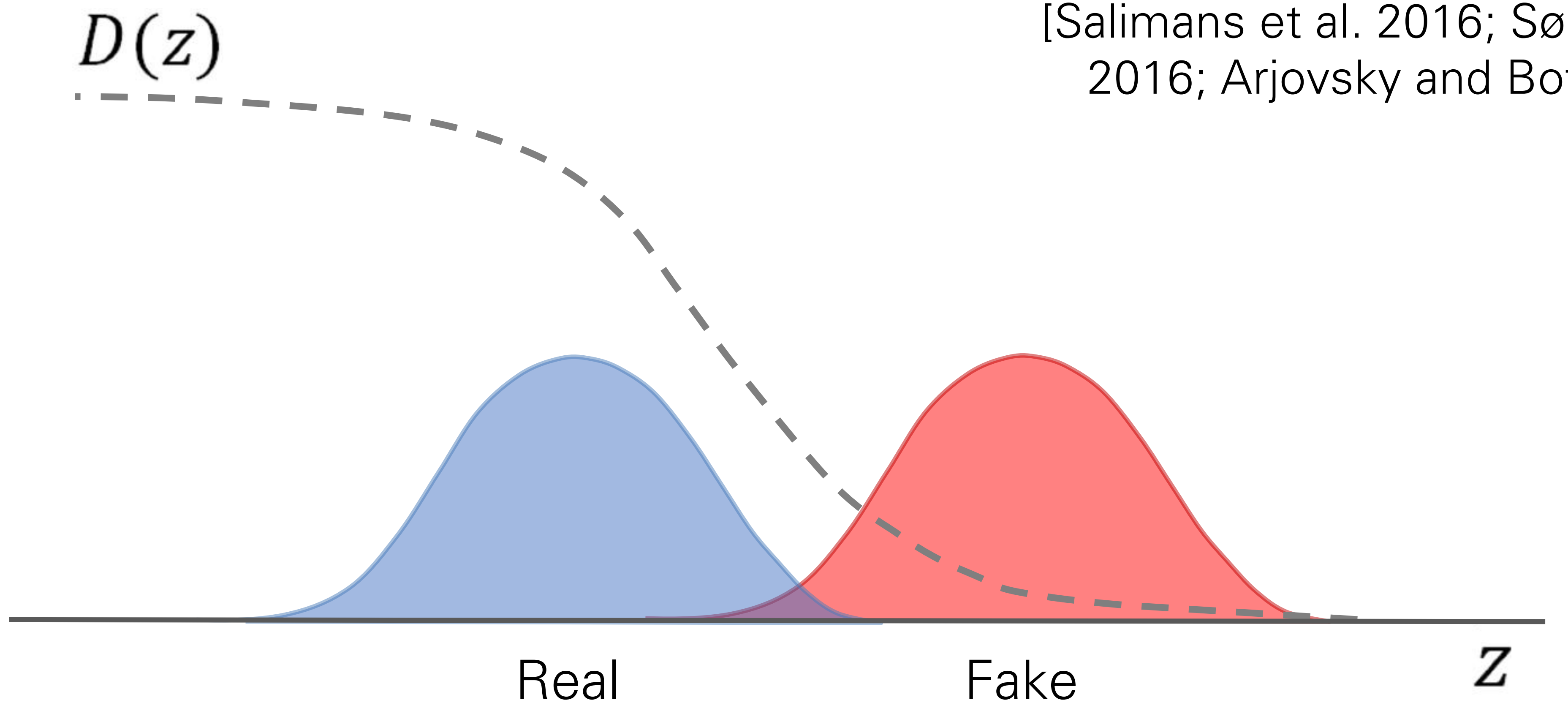
Variational Information Bottleneck

Instance Noise
[Salimans et al. 2016; Sønderby et al. 2016; Arjovsky and Bottou 2017]



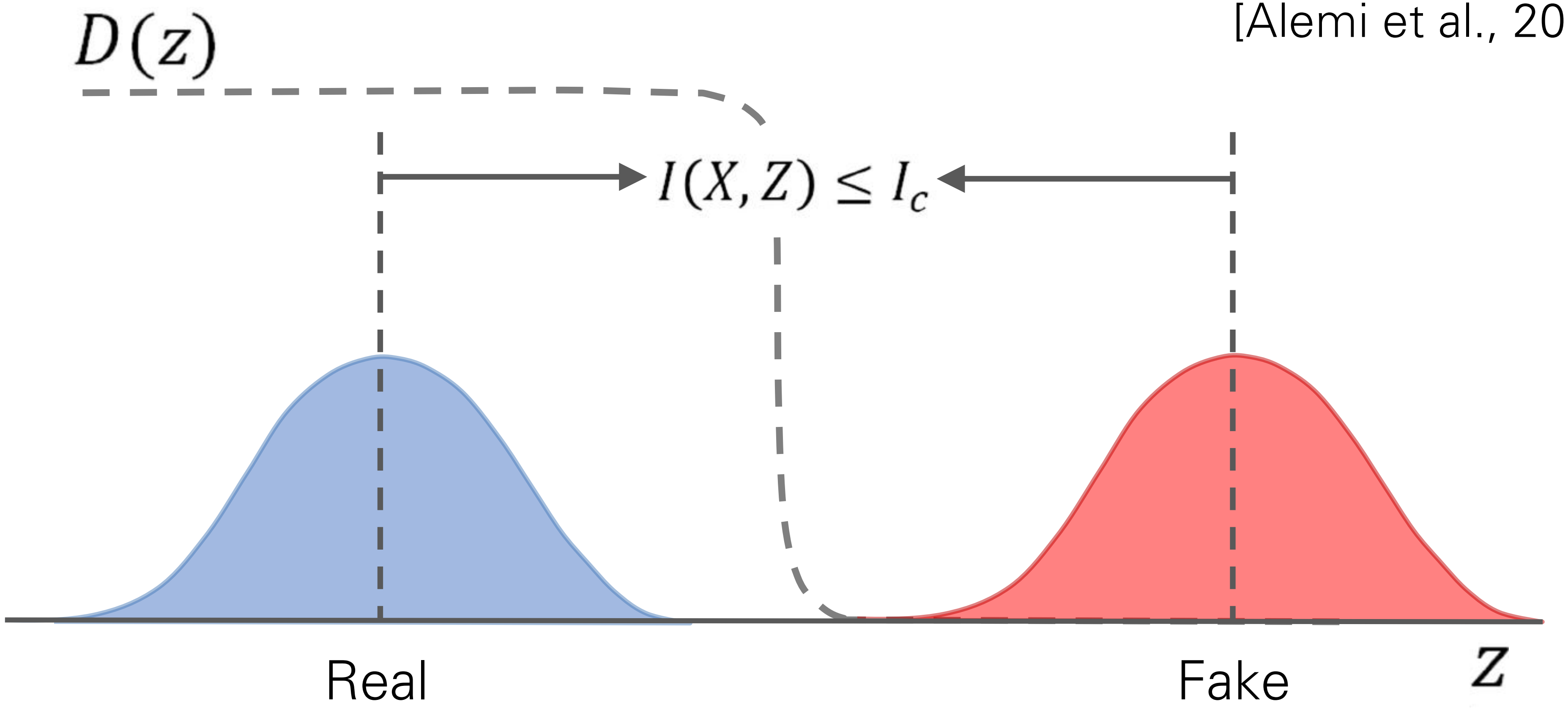
Variational Information Bottleneck

Instance Noise
[Salimans et al. 2016; Sønderby et al. 2016; Arjovsky and Bottou 2017]



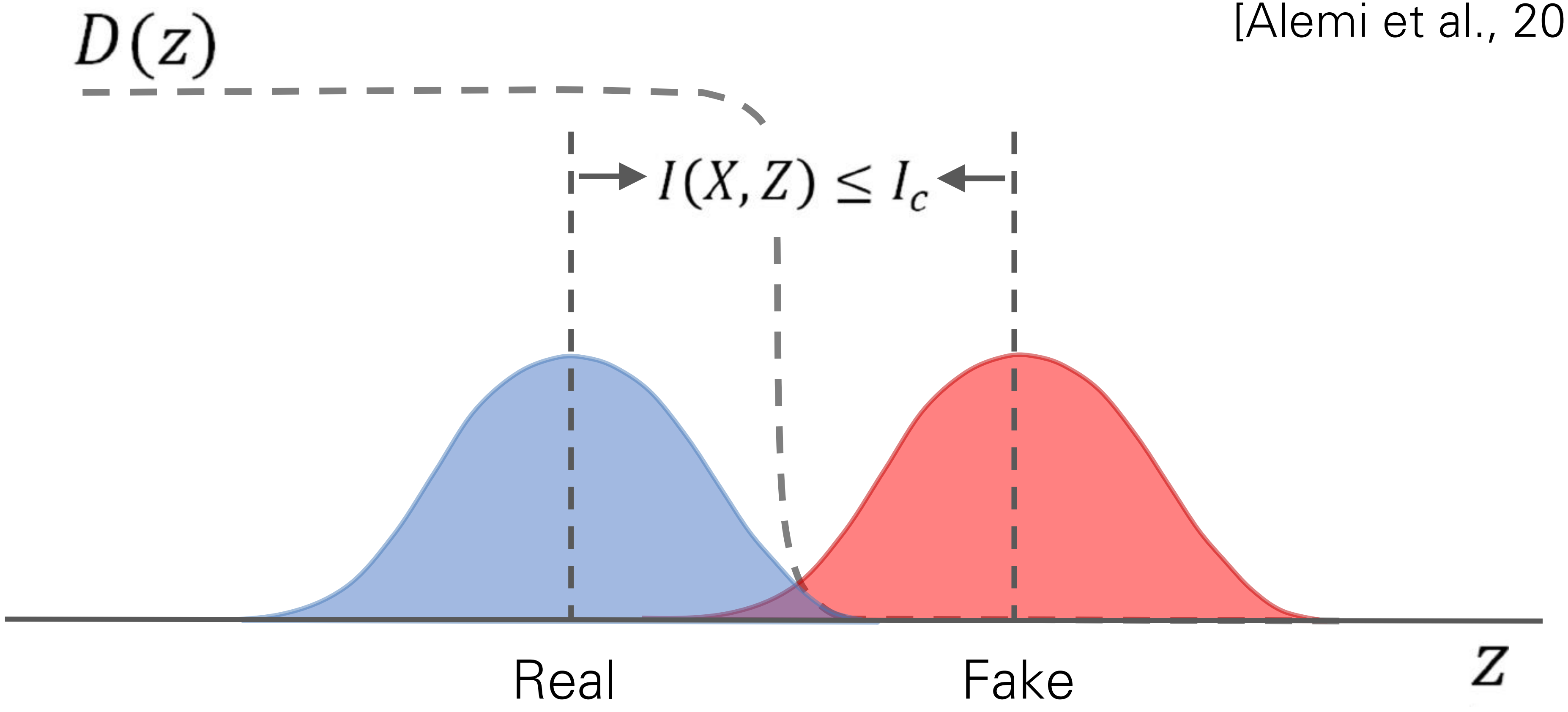
Variational Information Bottleneck

Variational Information Bottleneck
[Alemi et al., 2016]



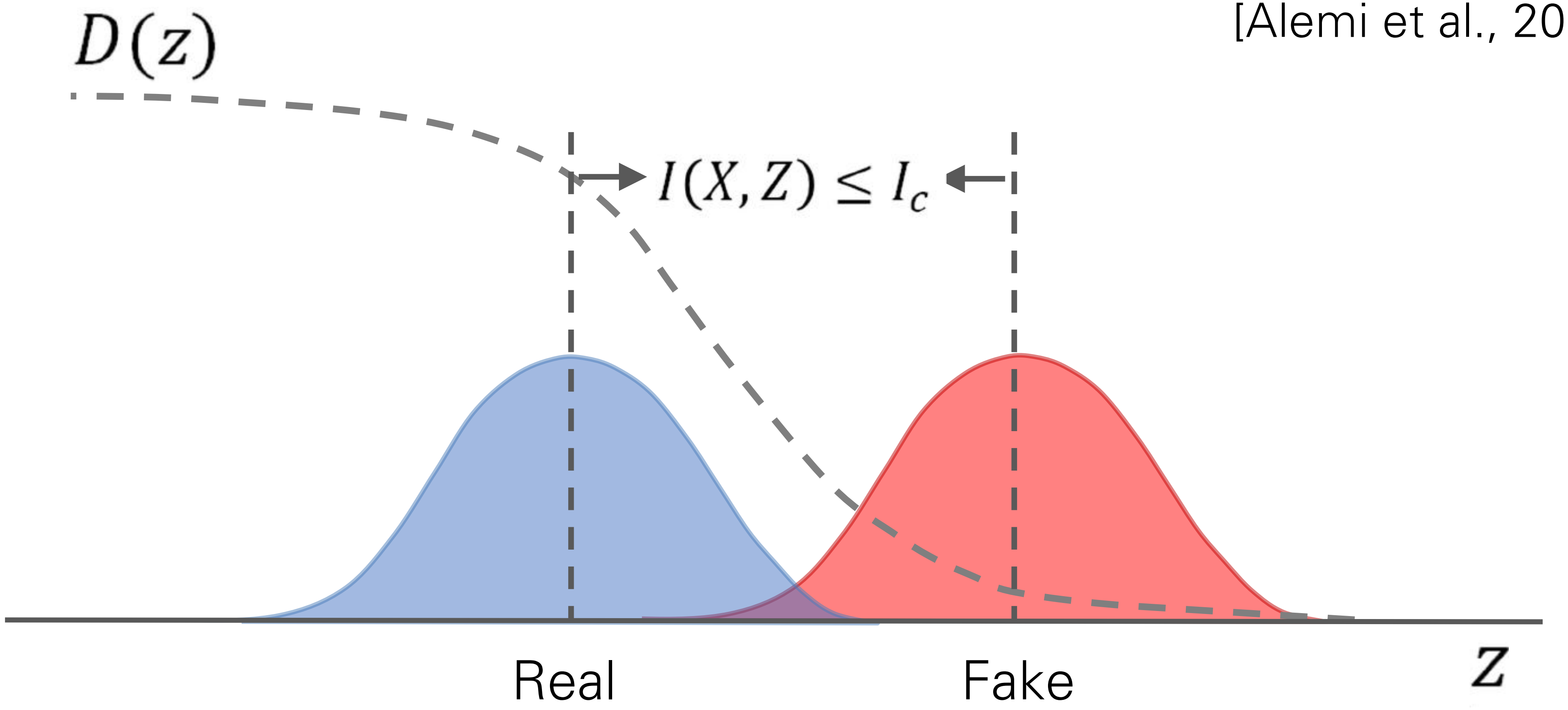
Variational Information Bottleneck

Variational Information Bottleneck
[Alemi et al., 2016]



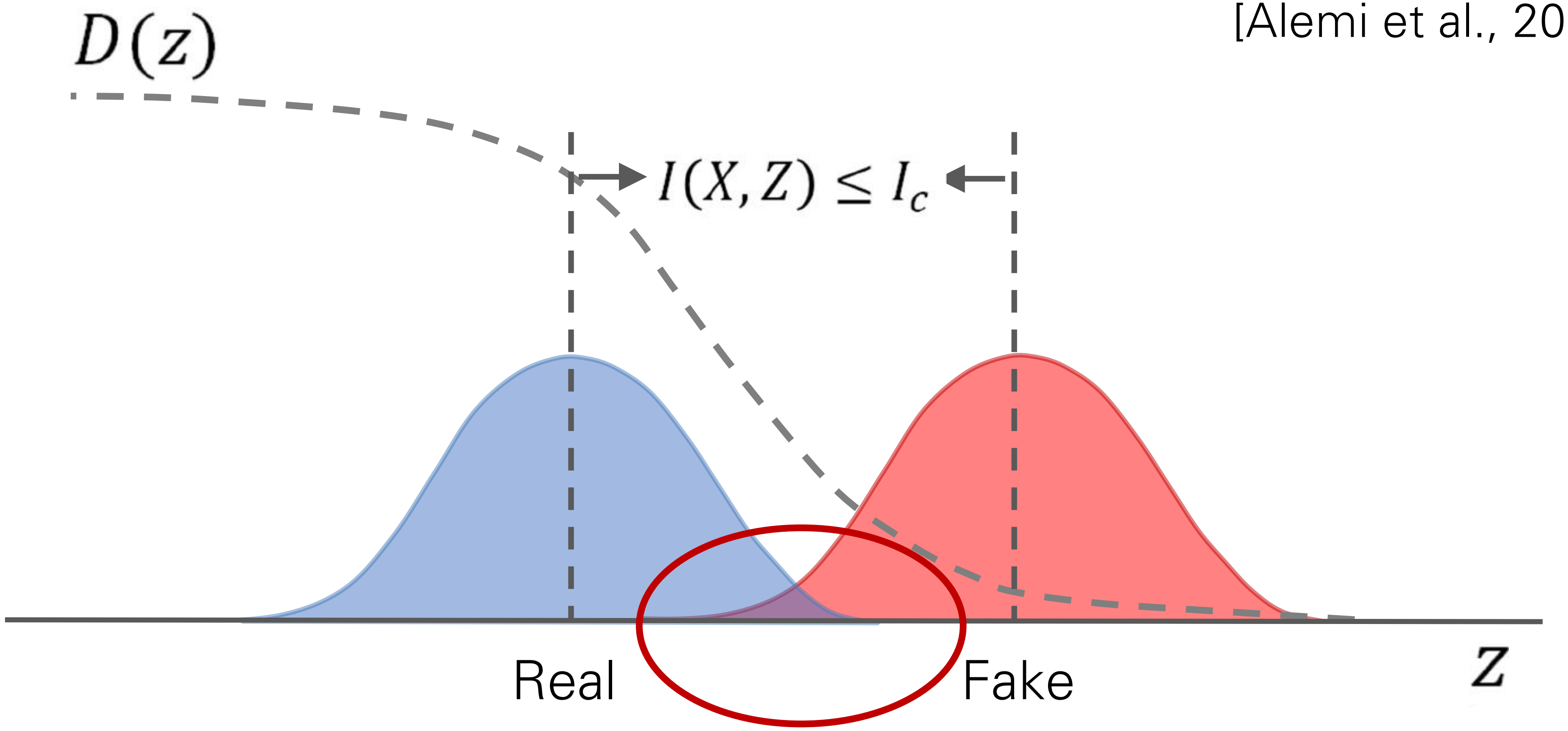
Variational Information Bottleneck

Variational Information Bottleneck
[Alemi et al., 2016]

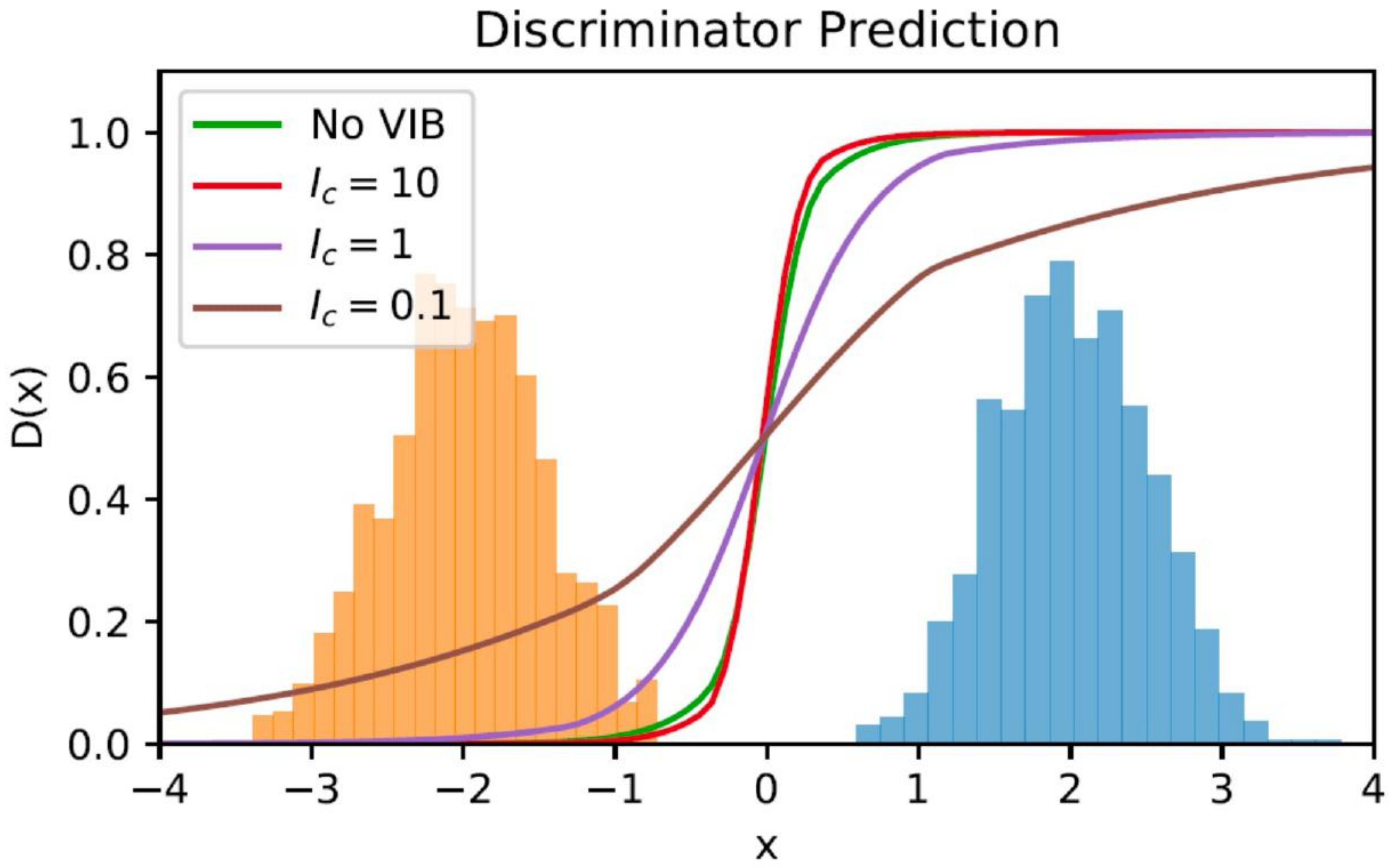


Variational Information Bottleneck

Variational Information Bottleneck
[Alemi et al., 2016]



Variational Information Bottleneck GAN

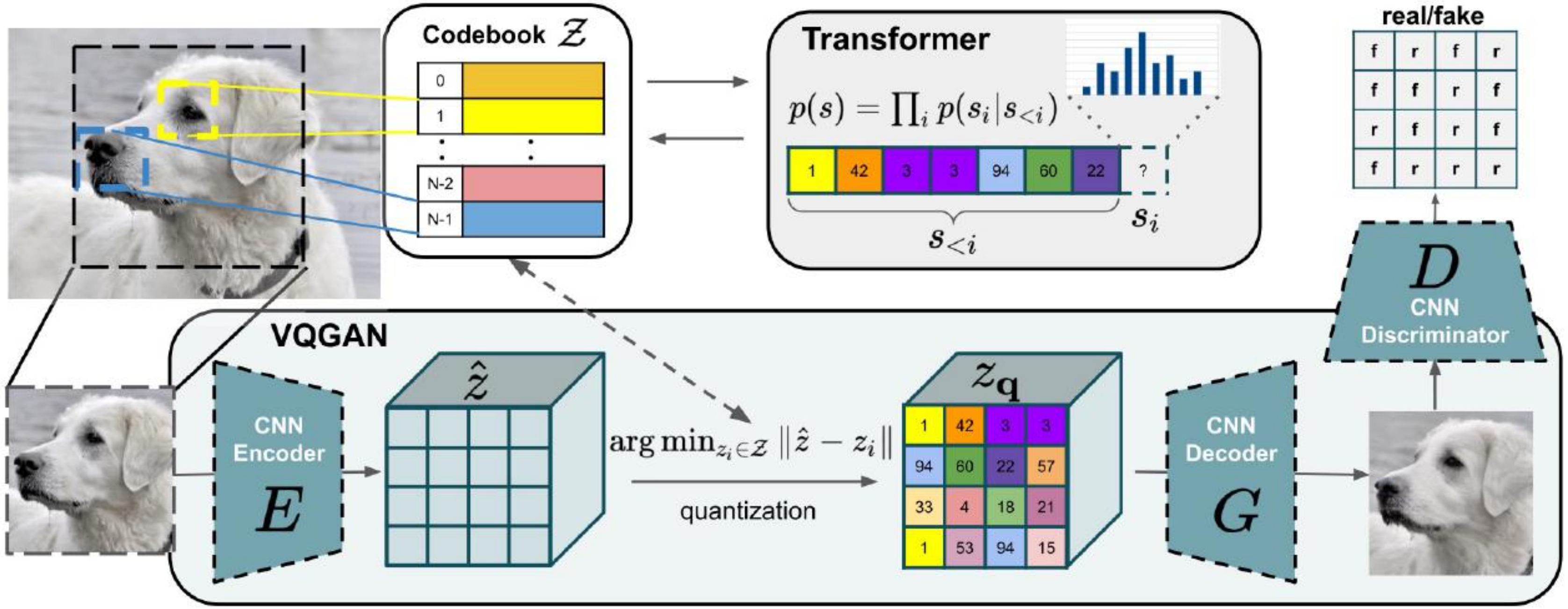


$$J(D,E) = \min_{D,E} \quad \mathbb{E}_{x\sim p^*(\mathbf{x})}\left[\mathbb{E}_{\mathbf{z}\sim E(\mathbf{z}|\mathbf{x})}[-\log(D(\mathbf{z}))]\right] + \mathbb{E}_{\mathbf{x}\sim G(\mathbf{x})}\left[\mathbb{E}_{\mathbf{z}\sim E(\mathbf{z}|\mathbf{x})}[-\log(1-D(\mathbf{z}))]\right]$$
$$\text{s.t.} \quad \mathbb{E}_{\mathbf{x}\sim\tilde{p}(\mathbf{x})}[\mathrm{KL}[E(\mathbf{z}\mid\mathbf{x})\|r(\mathbf{z})]] \leq I_c$$

Lecture overview

- Motivation and Definition of Implicit Models
- Original GAN (Goodfellow et al, 2014)
- Evaluation: Parzen, Inception, Frechet
- Theory of GANs
- **GAN Progression**
  - DC GAN (Radford et al, 2016)
  - Improved Training of GANs (Salimans et al'16), Projected GAN (Sauer et al'21) WGAN, WGAN-GP, Progressive GAN, SN-GAN, SAGAN
  - BigGAN, BigGAN-Deep, StyleGAN, StyleGAN2, StyleGAN3, StyleGAN-XL, Self-Distilled StyleGAN, VIB-GAN, **VQ-GAN**
- Conditional GANs
- GANs and Representations
- Applications

VQGAN



The complete objective for finding the optimal compression model $\mathcal{Q}^* = \{E^*, G^*, \mathcal{Z}^*\}$ then reads

$$\mathcal{Q}^* = \underset{E,G,\mathcal{Z}}{\arg\min}\, \underset{D}{\max}\, \mathbb{E}_{x\sim p(x)} \Big[\mathcal{L}_{\mathrm{VQ}}(E, G, \mathcal{Z}) + \lambda \mathcal{L}_{\mathrm{GAN}}(\{E, G, \mathcal{Z}\}, D) \Big], \quad (6)$$

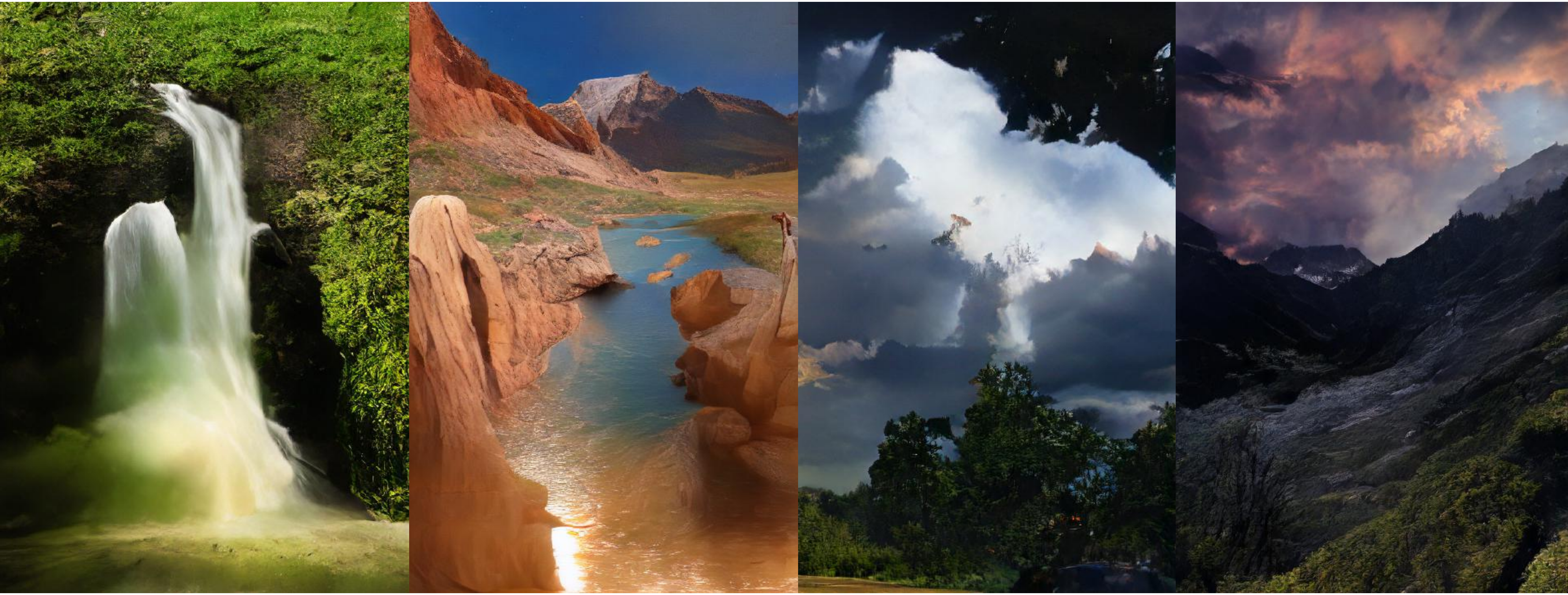
where we compute the adaptive weight λ according to

$$\lambda = \frac{\nabla_{G_L}[\mathcal{L}_{\mathrm{rec}}]}{\nabla_{G_L}[\mathcal{L}_{\mathrm{GAN}}] + \delta} \quad (7)$$

where $\mathcal{L}_{\mathrm{rec}}$ is the perceptual reconstruction loss [81], $\nabla_{G_L}[\cdot]$ denotes the gradient of its input w.r.t. the last layer L of

- A convolutional VQGAN to learn a codebook of context-rich visual parts
- An autoregressive Transformer to generate novel samples

S-FLCKR Samples from Semantic Layouts



ImageNet Samples

Quantitative Evaluation

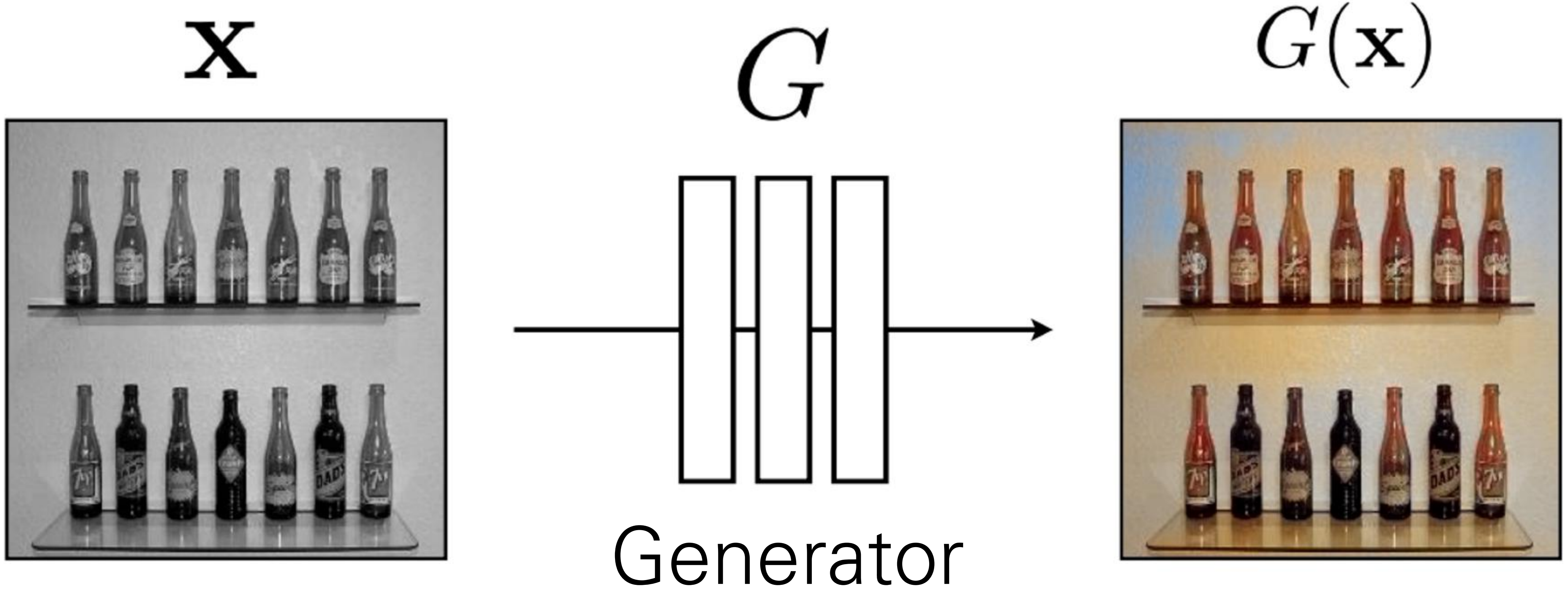
CelebA-HQ 256 × 256		FFHQ 256 × 256	
Method	FID ↓	Method	FID ↓
GLOW [33]	69.0	VDVAE ($t = 0.7$) [11]	38.8
NVAE [59]	40.3	VDVAE ($t = 1.0$)	33.5
PIONEER (B.) [21]	39.2 (25.3)	VDVAE ($t = 0.8$)	29.8
NCPVAE [1]	24.8	VDVAE ($t = 0.9$)	28.5
VAEBM [66]	20.4	*VQGAN*+P.SNAIL	21.9
Style ALAE [49]	19.2	BigGAN	12.4
DC-VAE [47]	15.8	**ours**	11.4
ours	10.7	U-Net GAN (+aug) [57]	10.9 (7.6)
PGGAN [27]	8.0	StyleGAN2 (+aug) [30]	3.8 (3.6)

Table 3. FID score comparison for face image synthesis. CelebA-HQ results reproduced from [1, 47, 66, 22], FFHQ from [57, 28].

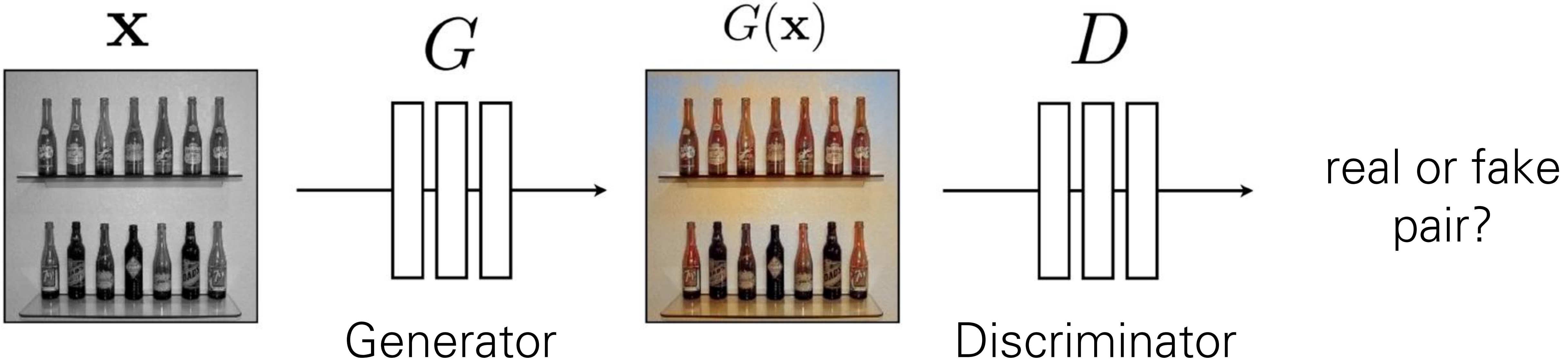
Lecture overview

- Motivation and Definition of Implicit Models
- Original GAN (Goodfellow et al, 2014)
- Evaluation: Parzen, Inception, Frechet
- Theory of GANs
- GAN Progression
- **Conditional GANs**, Cycle-Consistent Adversarial Networks
- GANs and Representations
- Applications

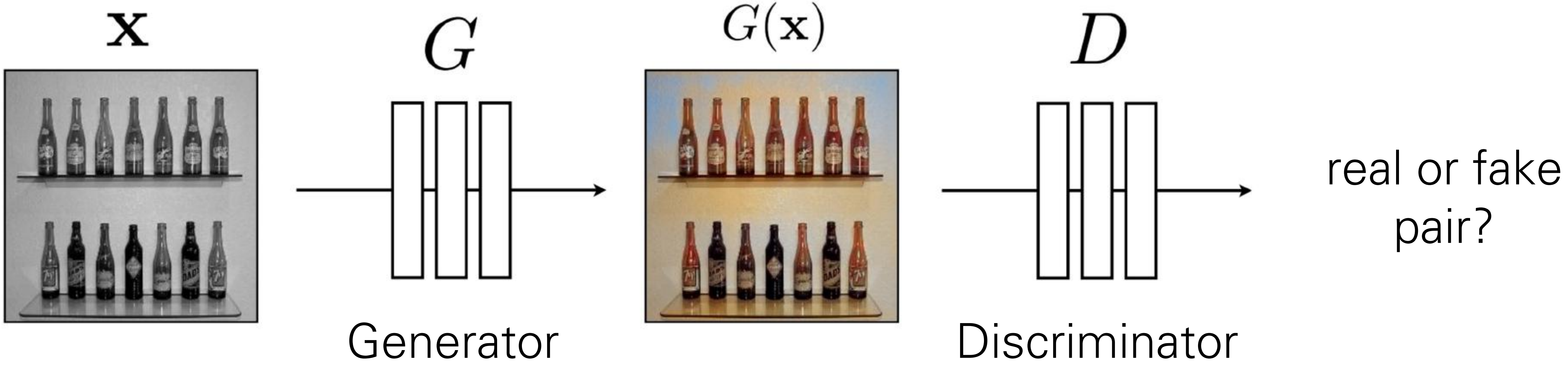
Conditional GANs / pix2pix



Conditional GANs / pix2pix



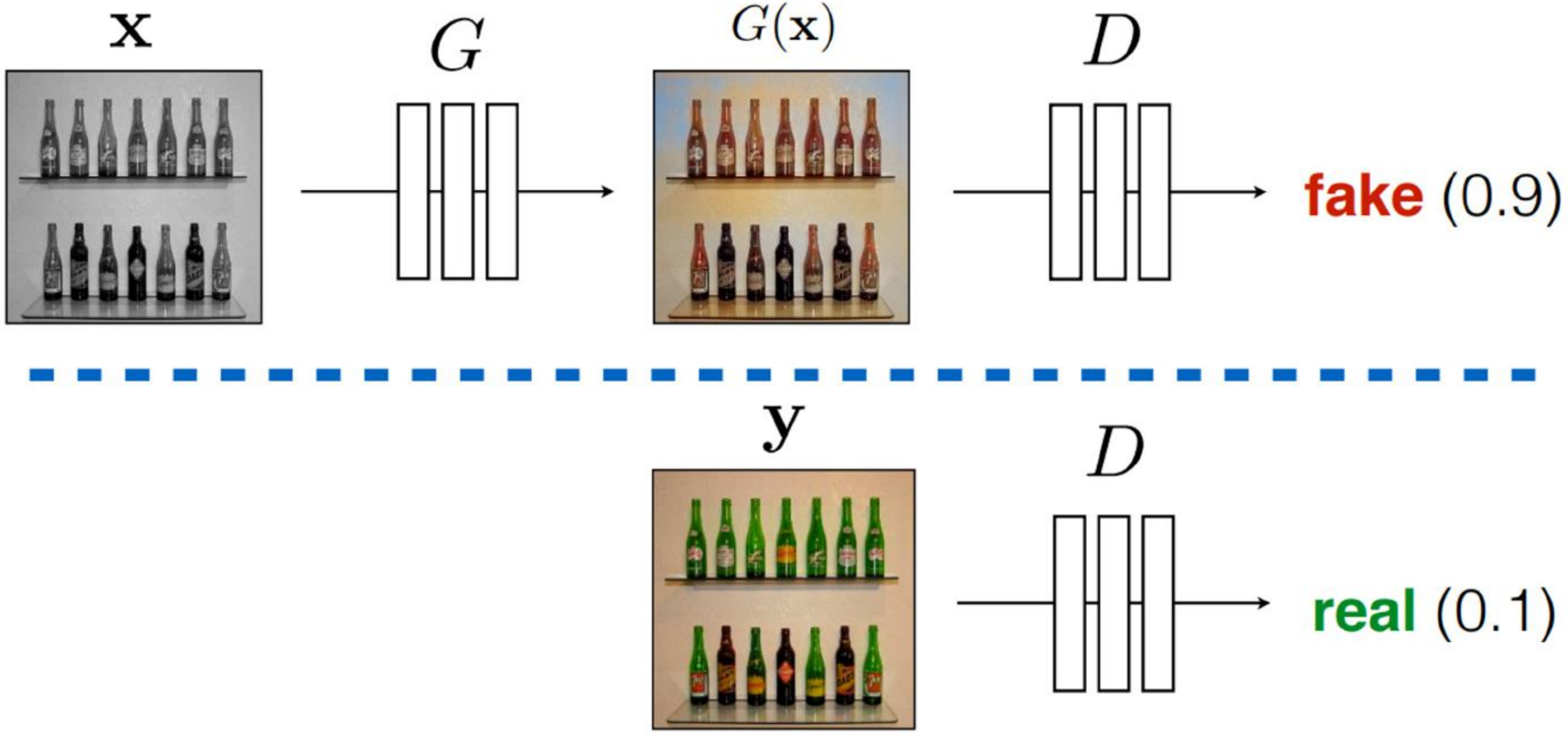
Conditional GANs / pix2pix



G tries to synthesize fake images that fool **D**

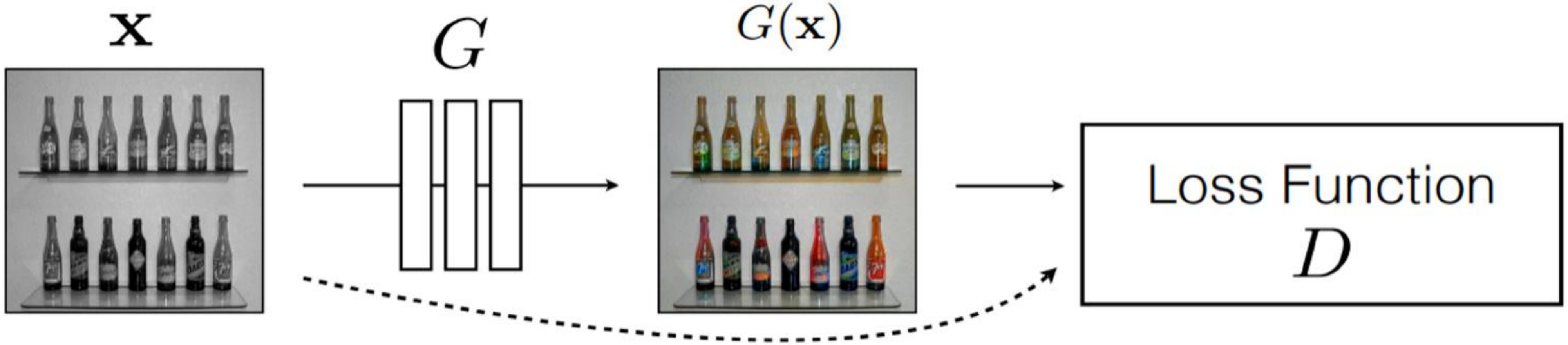
D tries to identify the fakes

Conditional GANs / pix2pix



$$\arg\max_{D} \ \mathbb{E}_{\mathbf{x},\mathbf{y}}[\ \boxed{\log D(G(\mathbf{x}))} \ + \ \boxed{\log(1 - D(\mathbf{y}))}\ ]$$

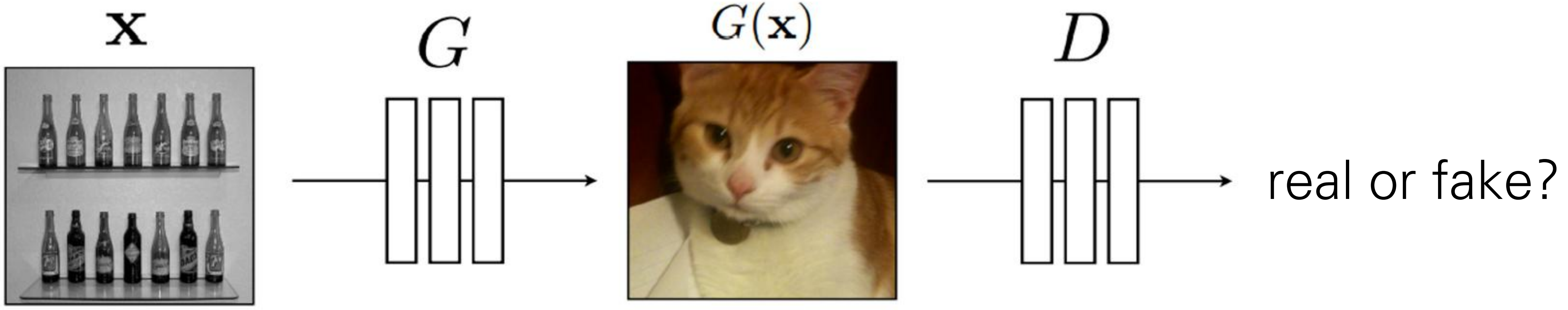
Conditional GANs / pix2pix



G's perspective: **D** is a loss function.

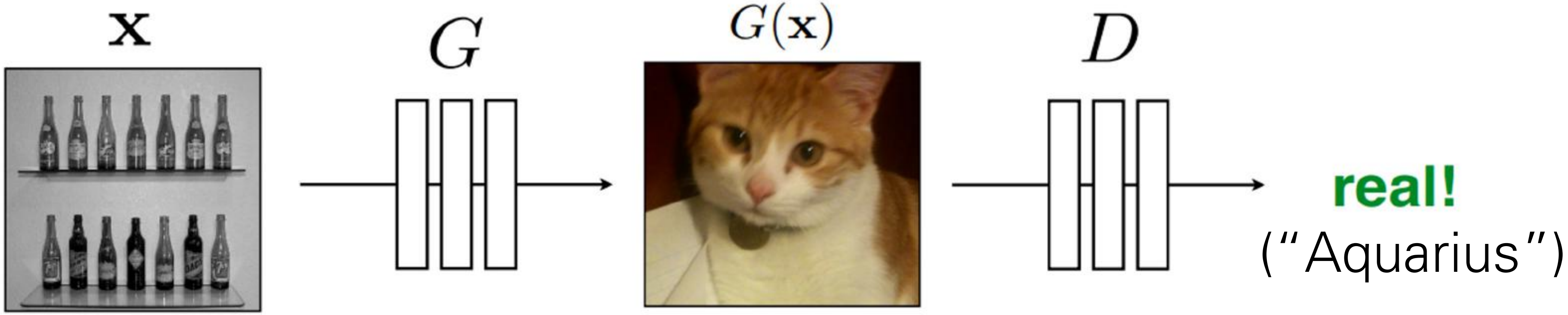
Rather than being hand-designed, it is <u>learned</u>.

Conditional GANs / pix2pix



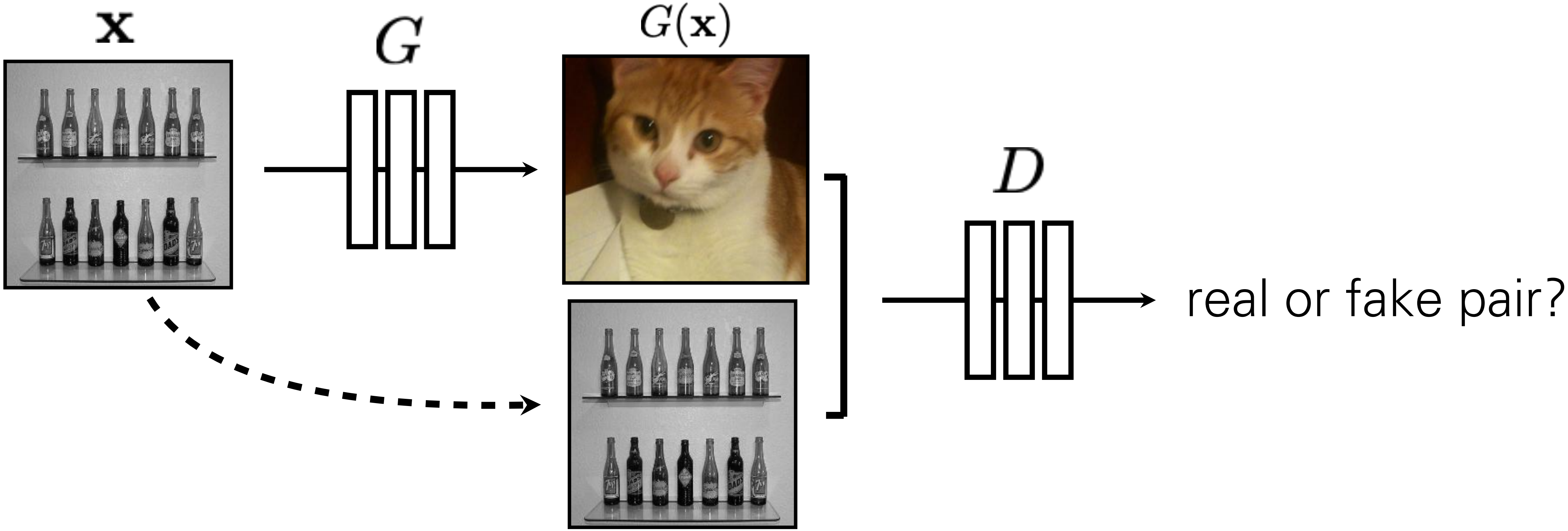
$$\arg\min_{G}\max_{D}\ \mathbb{E}_{\mathbf{x},\mathbf{y}}[\ \log D(G(\mathbf{x}))\ +\ \log(1-D(\mathbf{y}))\ ]$$

Conditional GANs / pix2pix



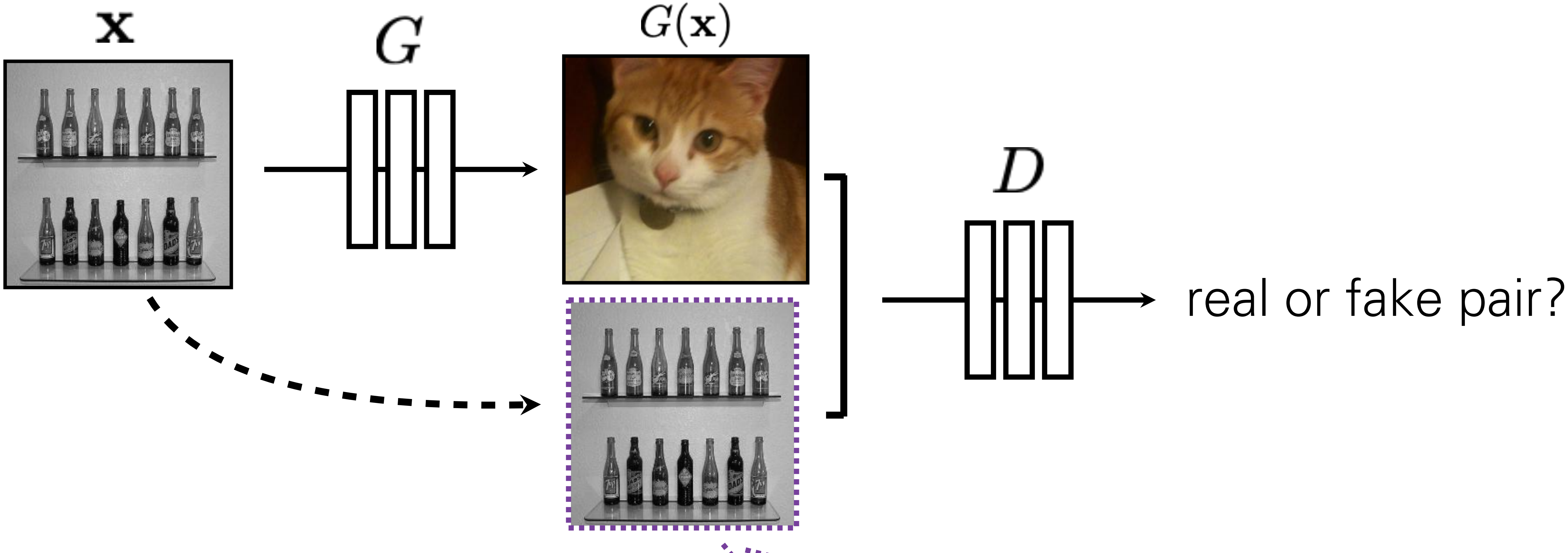
$$\arg\min_G \max_D \ \mathbb{E}_{\mathbf{x},\mathbf{y}}[\ \log D(G(\mathbf{x})) \quad + \quad \log(1 - D(\mathbf{y})) \ ]$$

Conditional GANs / pix2pix



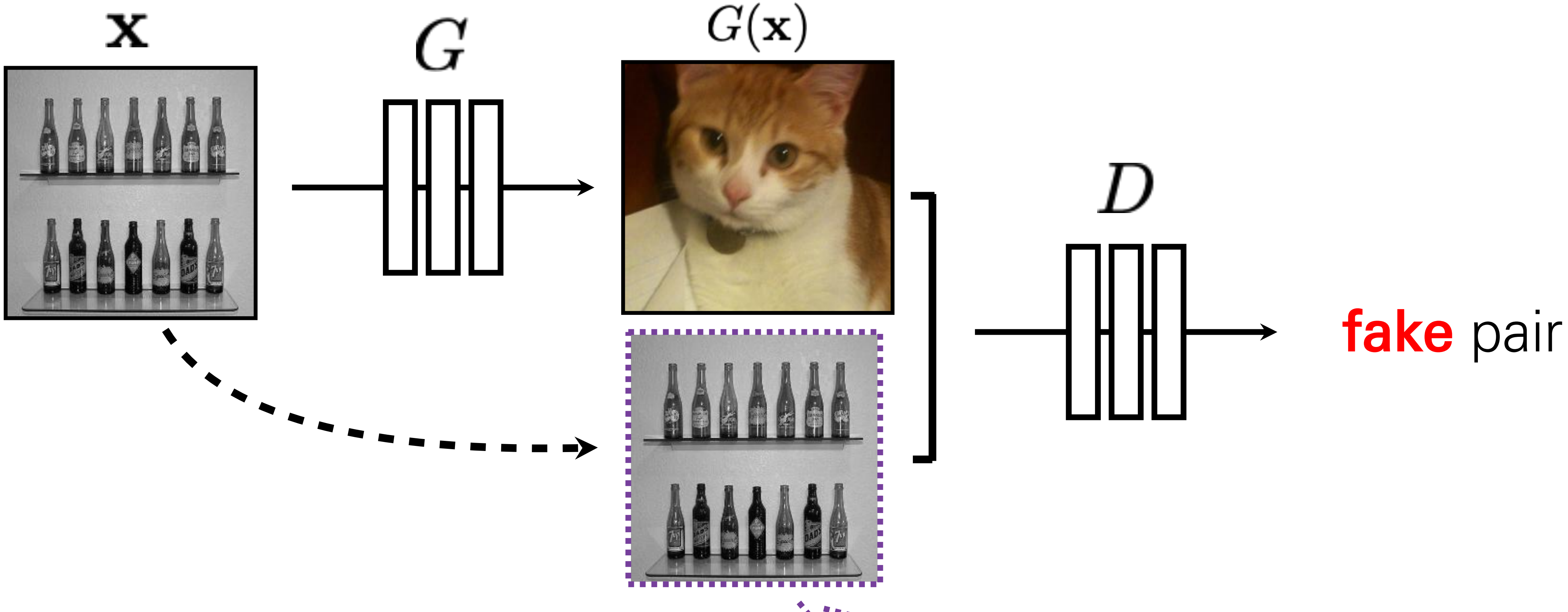
$$\arg\min_G \max_D \ \mathbb{E}_{\mathbf{x},\mathbf{y}}[\ \log D(G(\mathbf{x})) \ + \ \log(1 - D(\mathbf{y}))\ ]$$

Conditional GANs / pix2pix



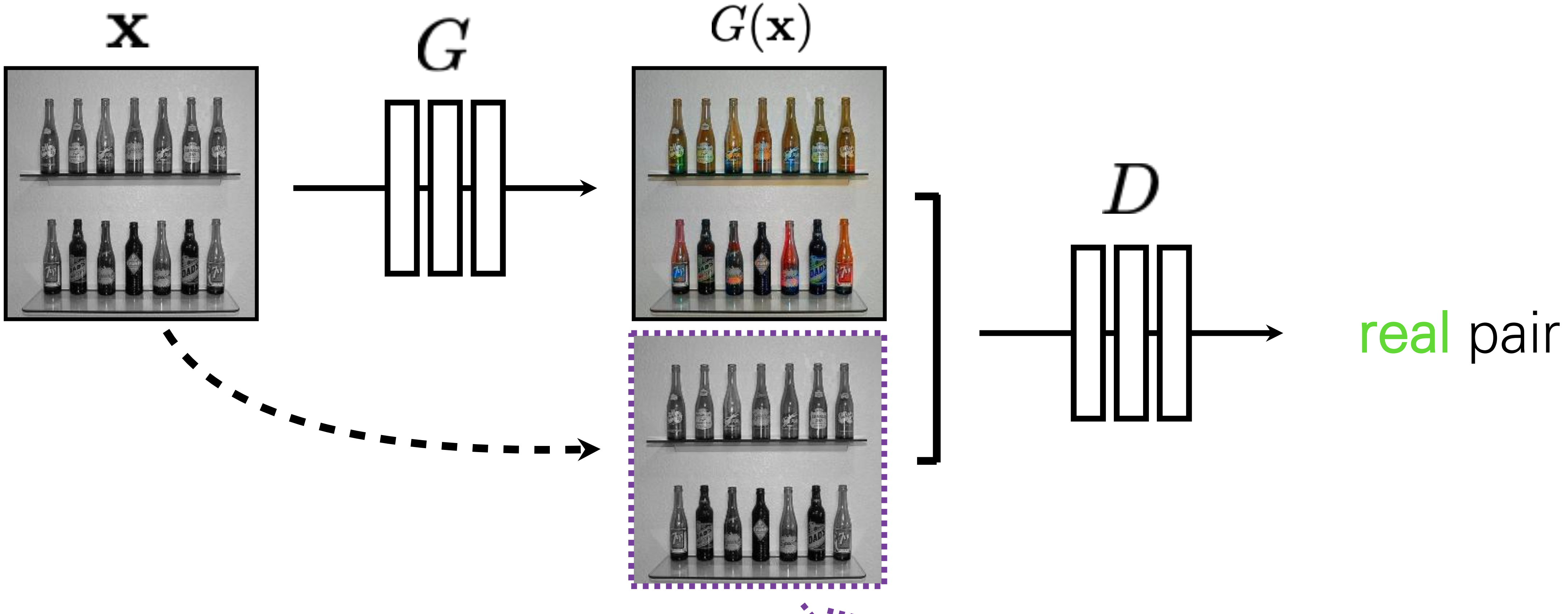
$$\arg\min_G \max_D \; \mathbb{E}_{\mathbf{x},\mathbf{y}}[\; \log D(\boxed{\mathbf{x}}, G(\mathbf{x})) + \log(1 - D(\boxed{\mathbf{x}}, \mathbf{y})) \;]$$

Conditional GANs / pix2pix



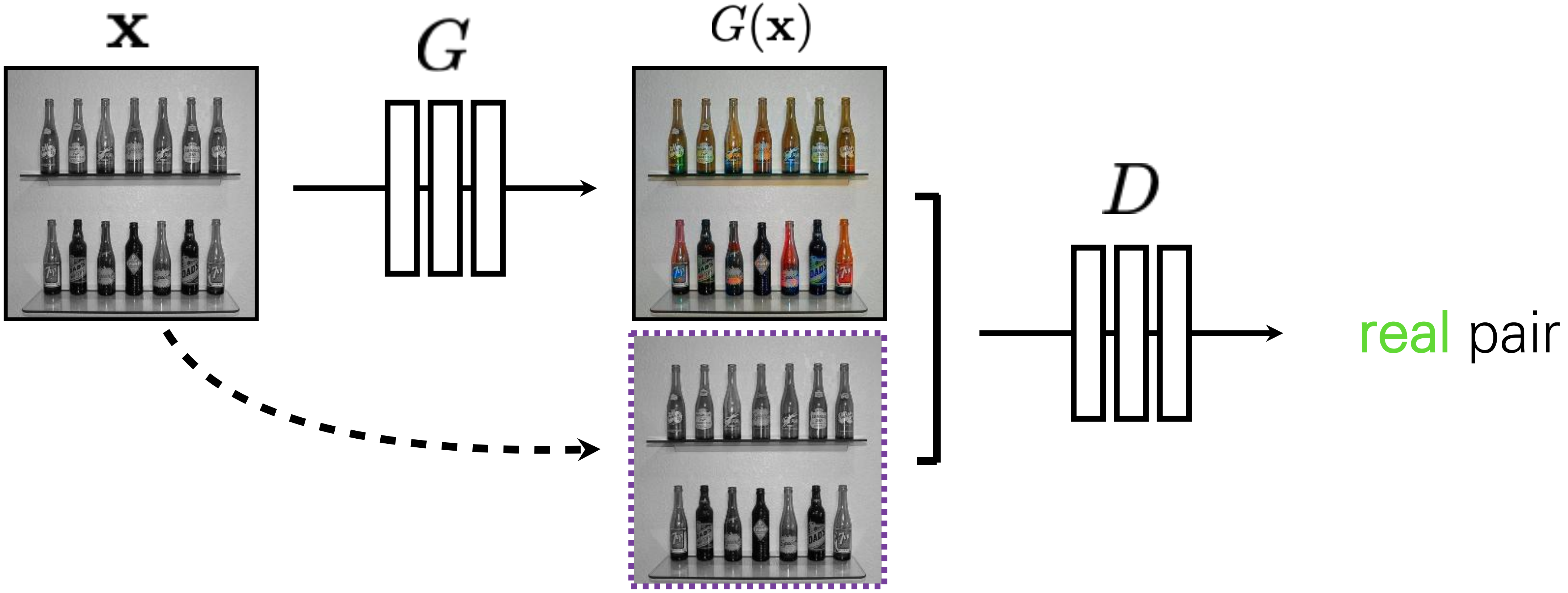
$$\arg\min_G \max_D \ \mathbb{E}_{\mathbf{x},\mathbf{y}}[\ \log D(\boxed{\mathbf{x}}, G(\mathbf{x})) + \log(1 - D(\boxed{\mathbf{x}}, \mathbf{y}))\ ]$$

Conditional GANs / pix2pix



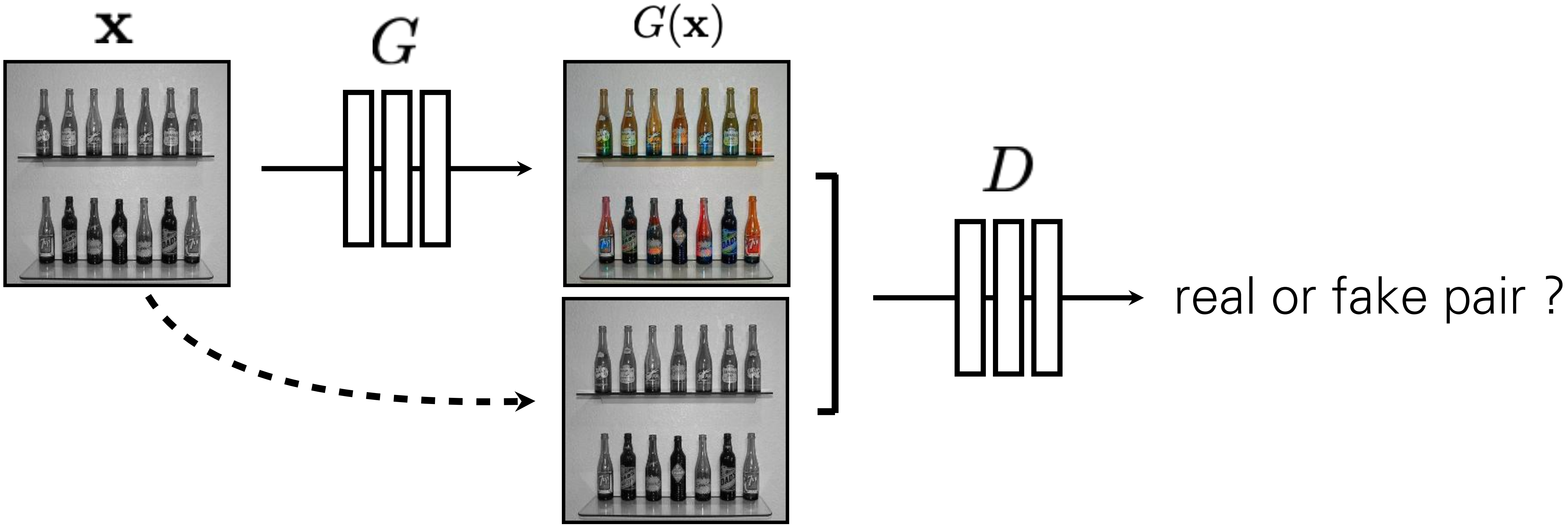
$$\arg\min_G \max_D \ \mathbb{E}_{\mathbf{x},\mathbf{y}}[\ \log D(\boxed{\mathbf{x}}, G(\mathbf{x})) + \log(1 - D(\boxed{\mathbf{x}}, \mathbf{y}))\ ]$$

Conditional GANs / pix2pix



$$\arg\min_G \max_D \; \mathbb{E}_{\mathbf{x},\mathbf{y}}[\; \log D(\boxed{\mathbf{x}}, G(\mathbf{x})) + \log(1 - D(\boxed{\mathbf{x}}, \mathbf{y})) \;]$$

Conditional GANs / pix2pix



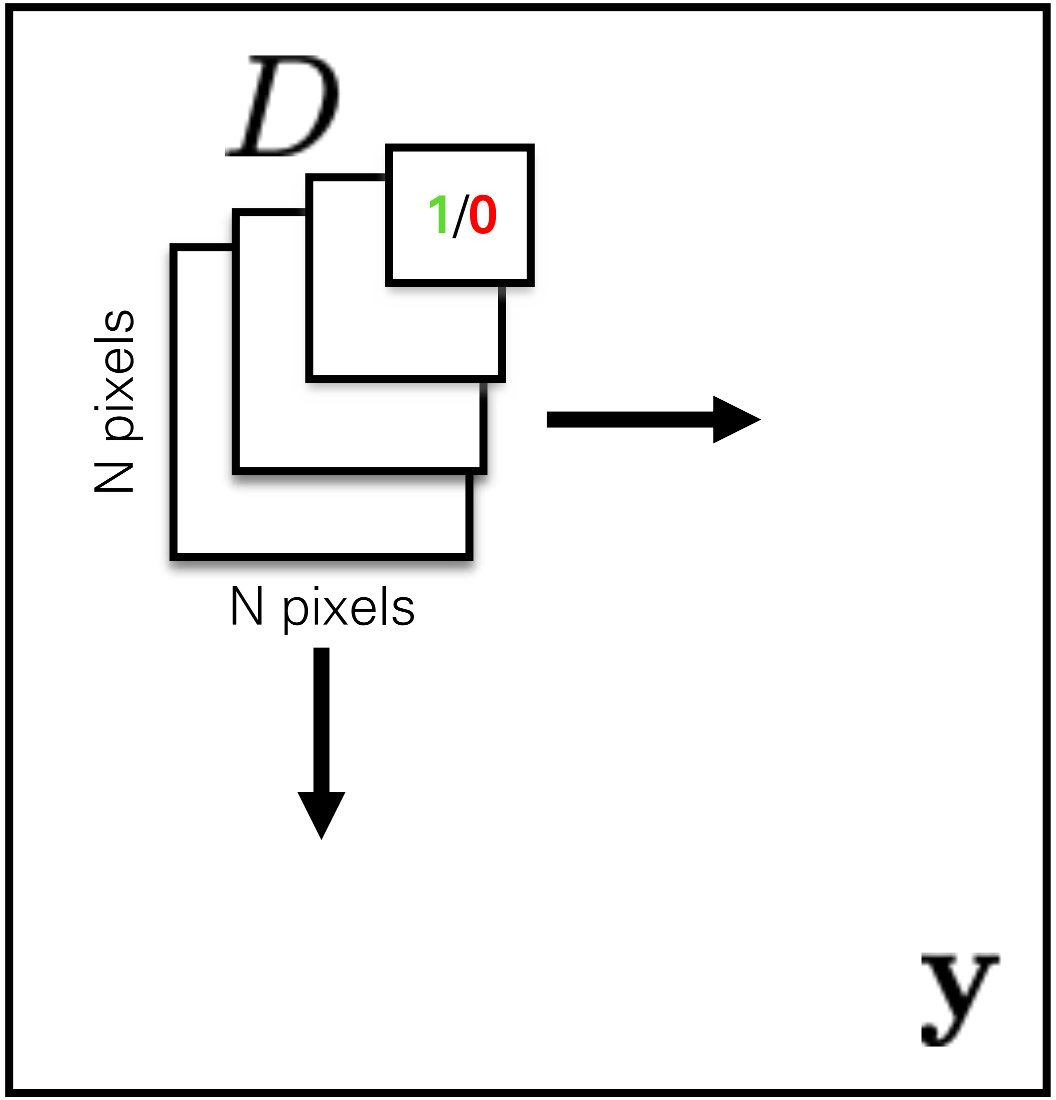
$$\arg\min_{G}\max_{D}\ \mathbb{E}_{\mathbf{x},\mathbf{y}}[\ \log D(\mathbf{x}, G(\mathbf{x})) + \log(1 - D(\mathbf{x}, \mathbf{y}))\ ]$$

Conditional GANs / pix2pix

$$G^* = \arg\min_G \max_D \mathcal{L}_{cGAN}(G, D) + \lambda \mathcal{L}_{L1}(G)$$

Conditional GANs / pix2pix

Shrinking the capacity: Patch Discriminator

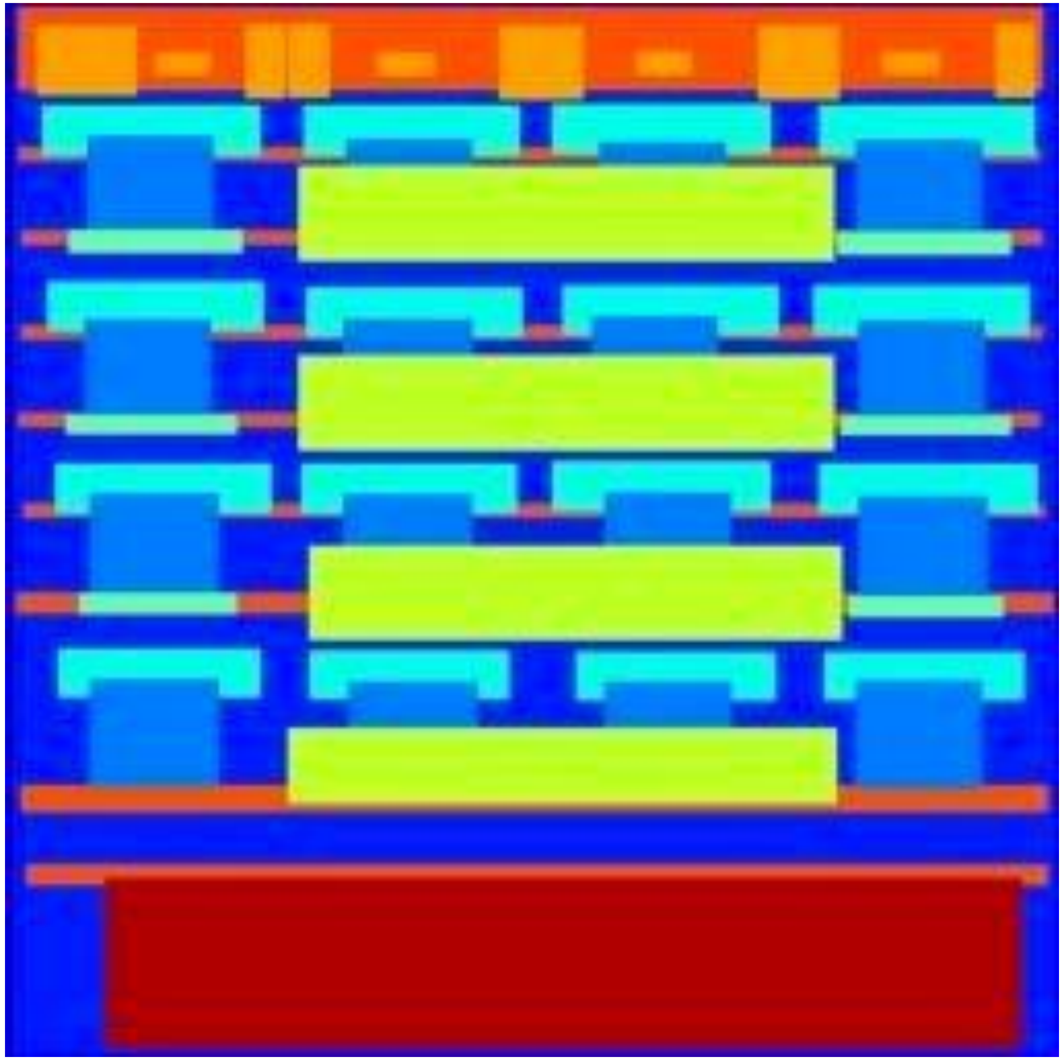


Rather than penalizing if output image looks fake, penalize if each overlapping patch in output looks fake

[Li & Wand 2016]
[Shrivastava et al. 2017]
[Isola et al. 2017]

Conditional GANs / pix2pix

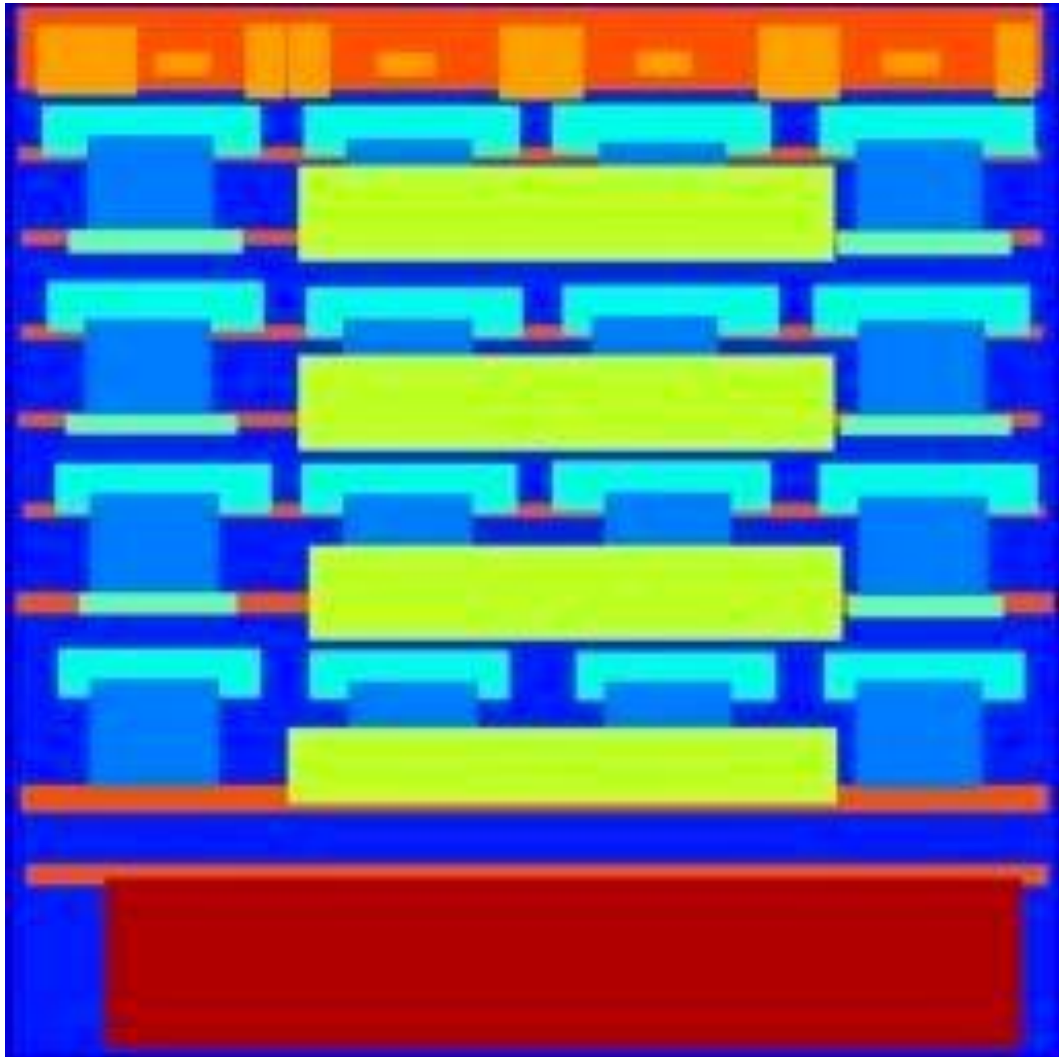
Input



1x1 Discriminator

Conditional GANs / pix2pix

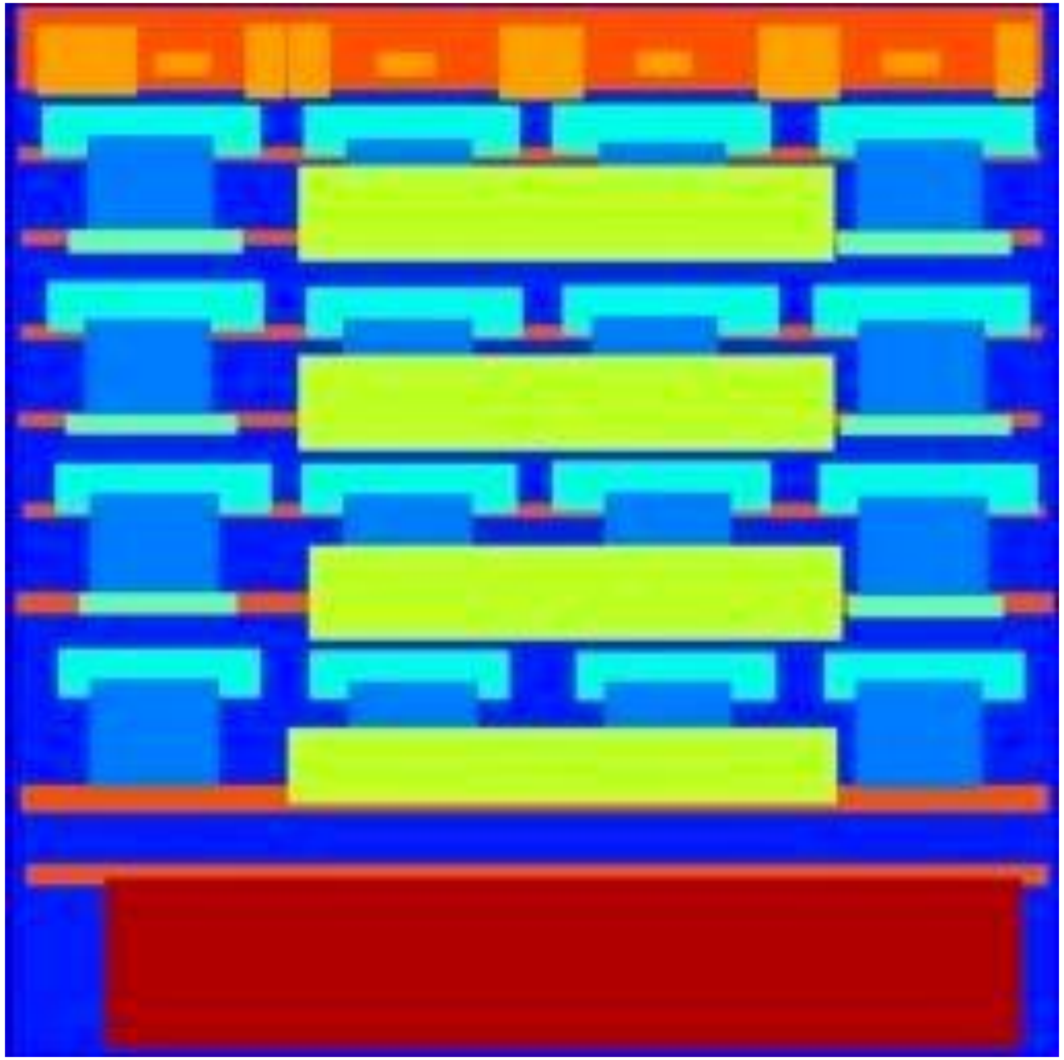
Input



16x16 Discriminator

Conditional GANs / pix2pix

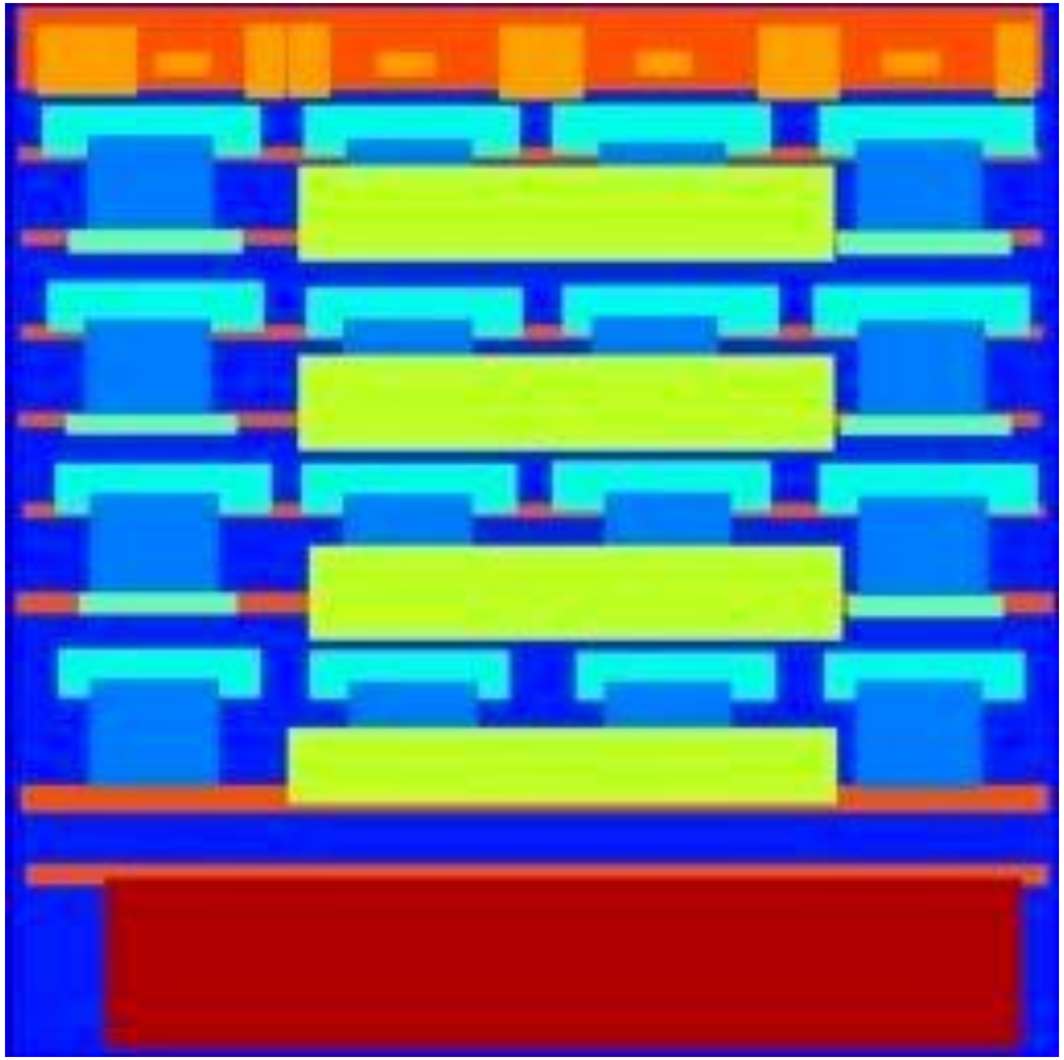
Input



70x70 Discriminator

Conditional GANs / pix2pix

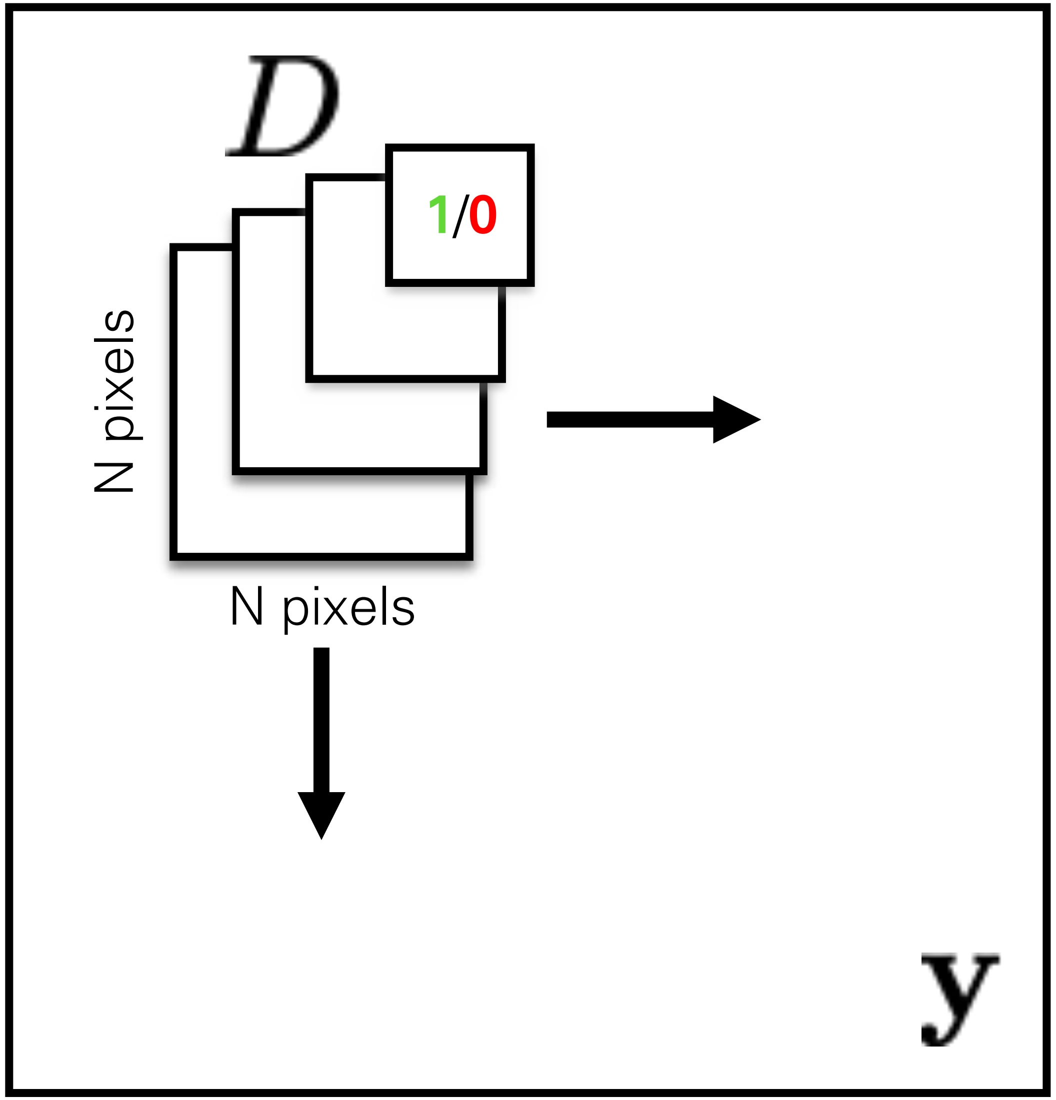
Input



16x16 Discriminator

Conditional GANs / pix2pix

Shrinking the capacity: Patch Discriminator

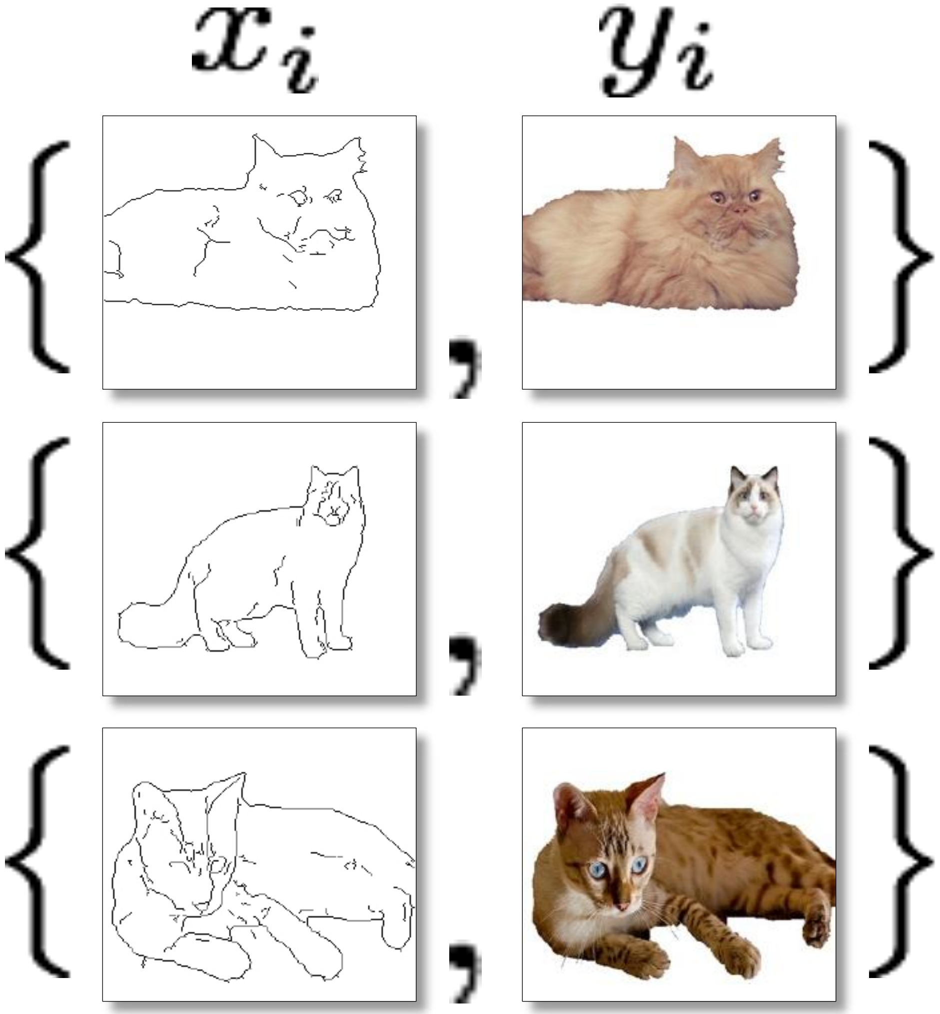


Rather than penalizing if output image looks fake, penalize if each overlapping patch in output looks fake

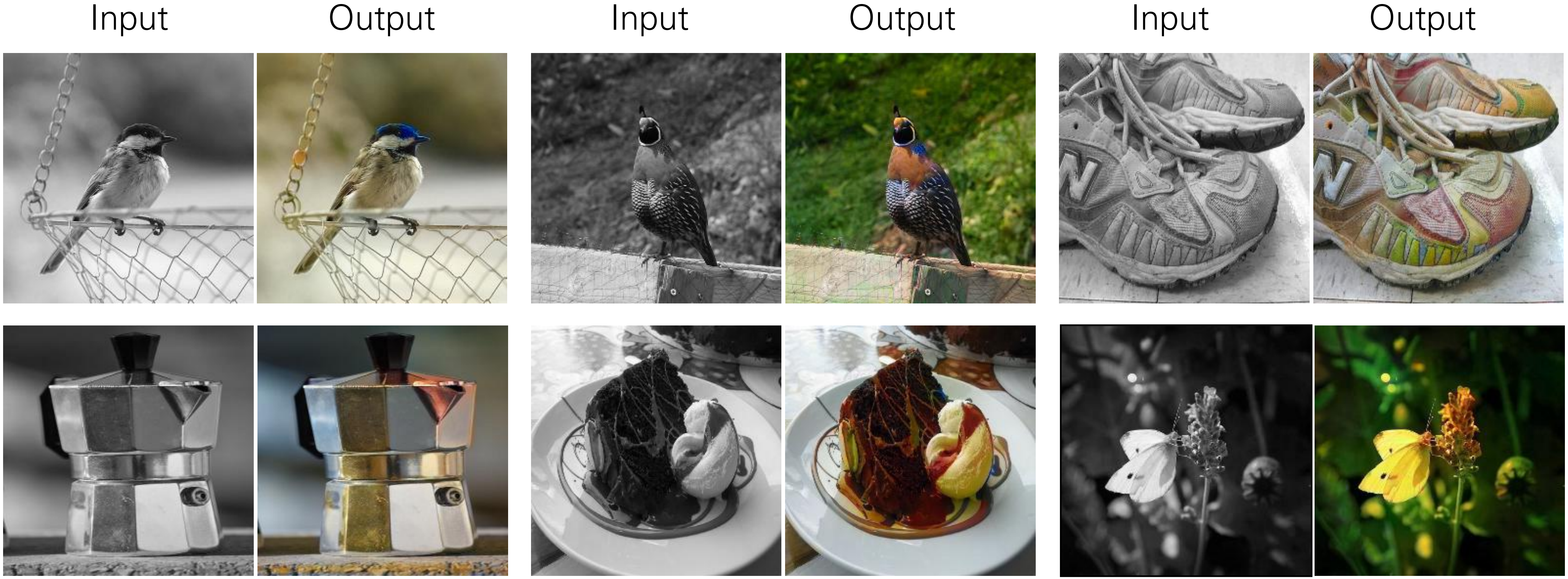
- Faster, fewer parameters
- More supervised observations
- Applies to arbitrarily large images

[Li & Wand 2016]
[Shrivastava et al. 2017]
[Isola et al. 2017]

Conditional GANs / pix2pix

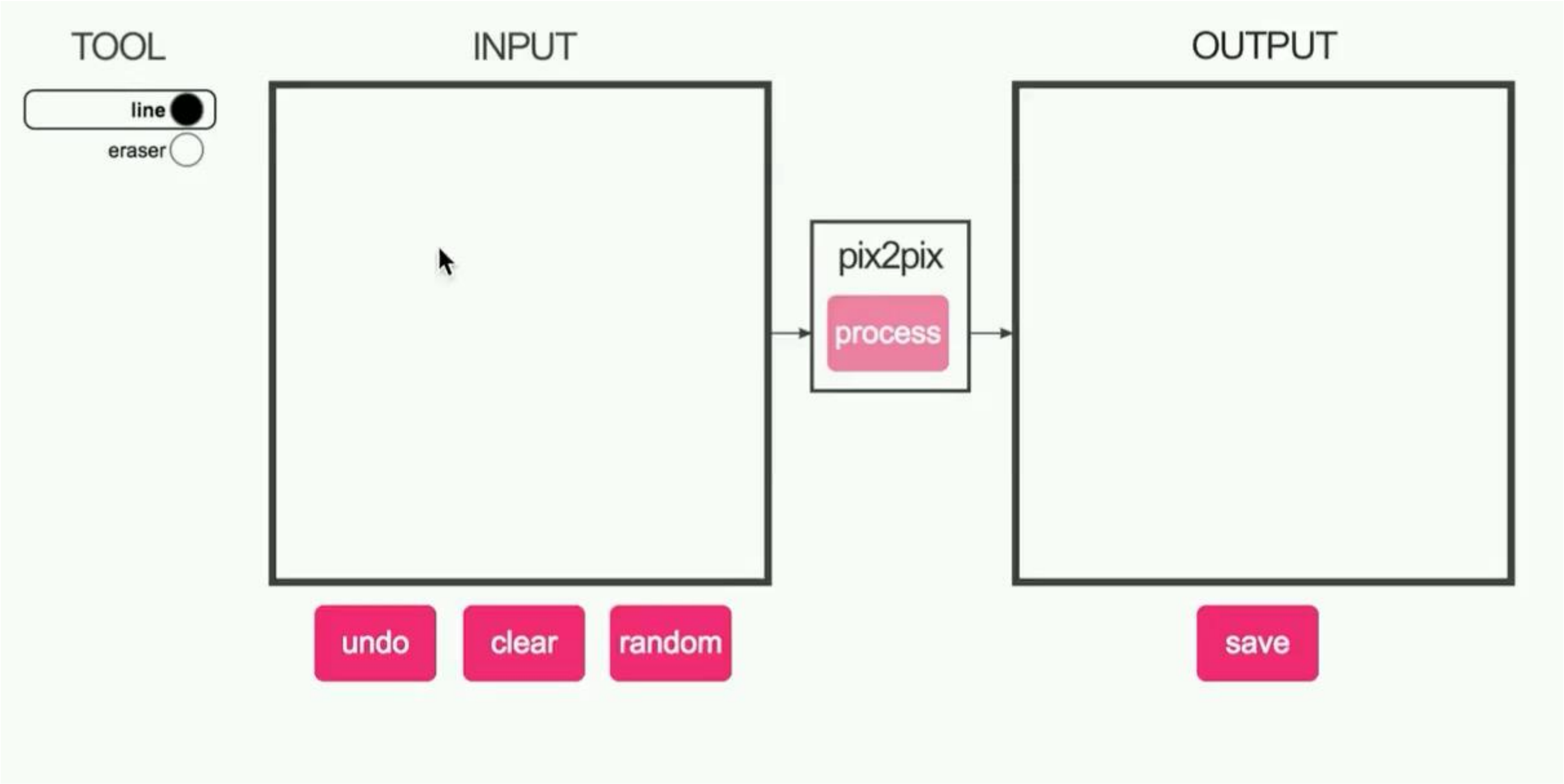


Conditional GANs / pix2pix   BW → Color

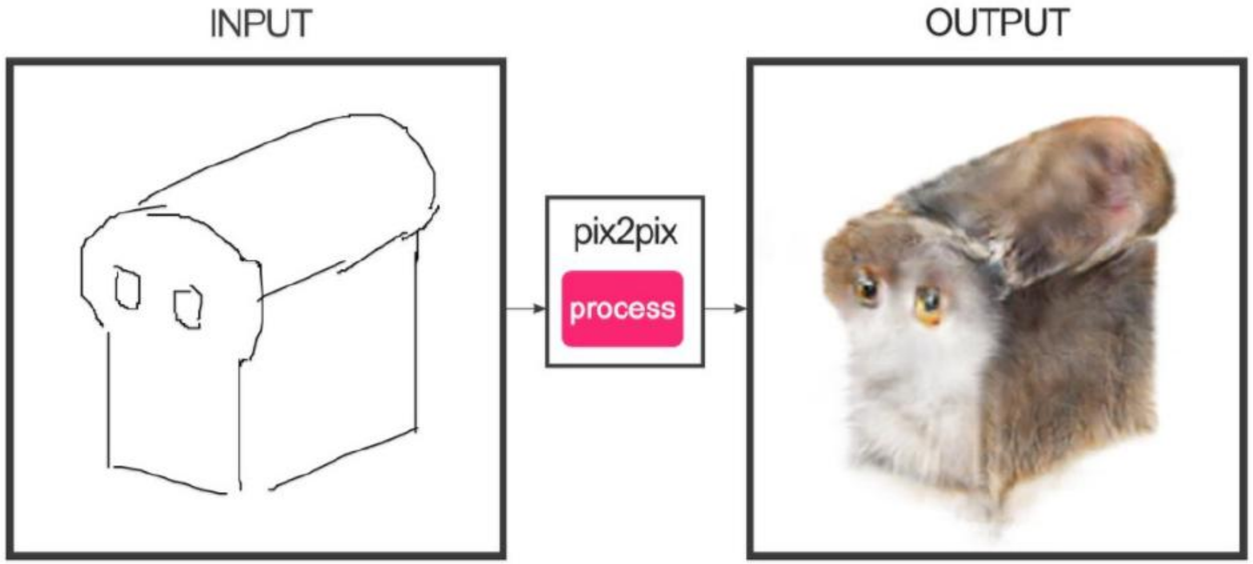


Data from [Russakovsky et al. 2015]

Conditional GANs / pix2pix #edges2cats [Chris Hesse]



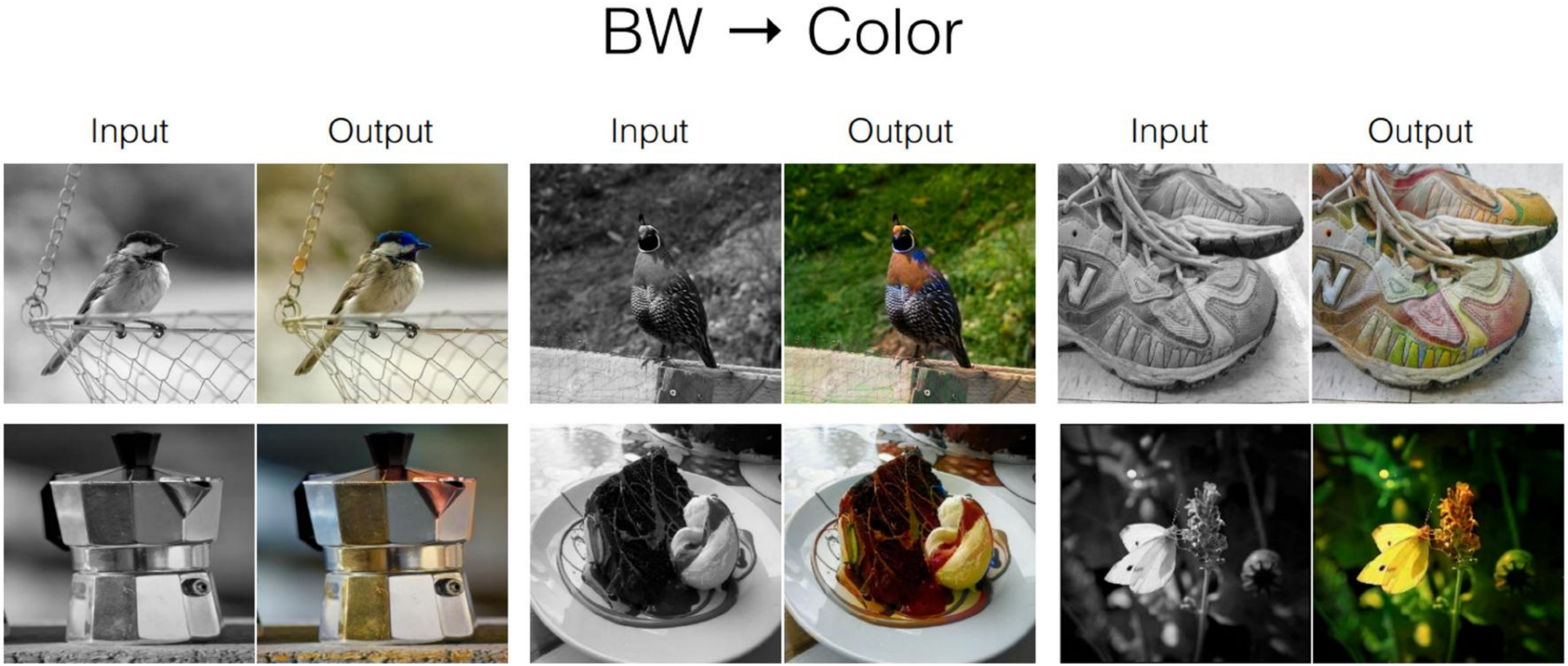
Conditional GANs / pix2pix



Ivy Tasi @ivymyt

Vitaly Vidmirov @vvid

Conditional GANs / pix2pix



Conditional GANs / pix2pix

Structured Prediction

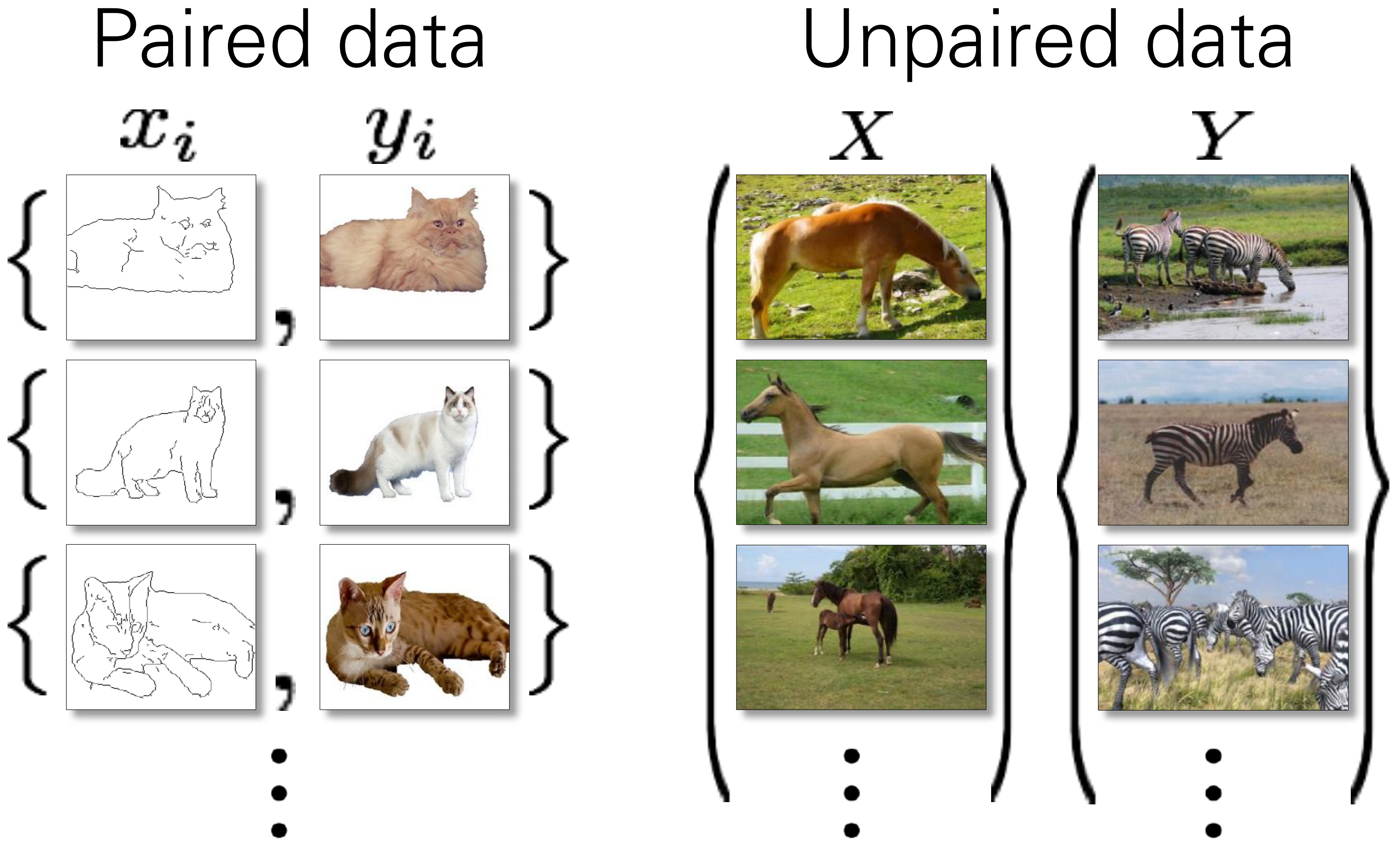
Input	Output	Target
$\mathbf{x}$	$\hat{\mathbf{y}}$	$\mathbf{y}$

$$L(\hat{\mathbf{y}}, \mathbf{y}) = ||\hat{\mathbf{y}} - \mathbf{y}||_2$$

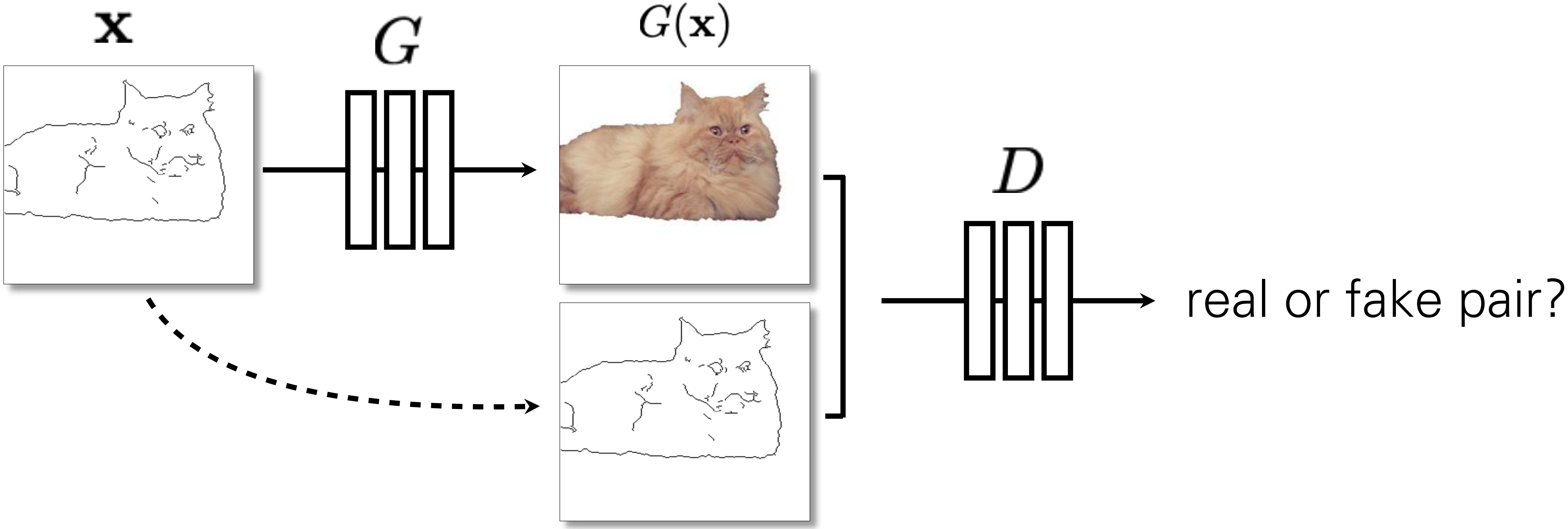
Lecture overview

- Motivation and Definition of Implicit Models
- Original GAN (Goodfellow et al, 2014)
- Evaluation: Parzen, Inception, Frechet
- Theory of GANs
- GAN Progression
- Conditional GANs, **Cycle-Consistent Adversarial Networks**
- GANs and Representations
- Applications

Cycle-Consistent Adversarial Networks

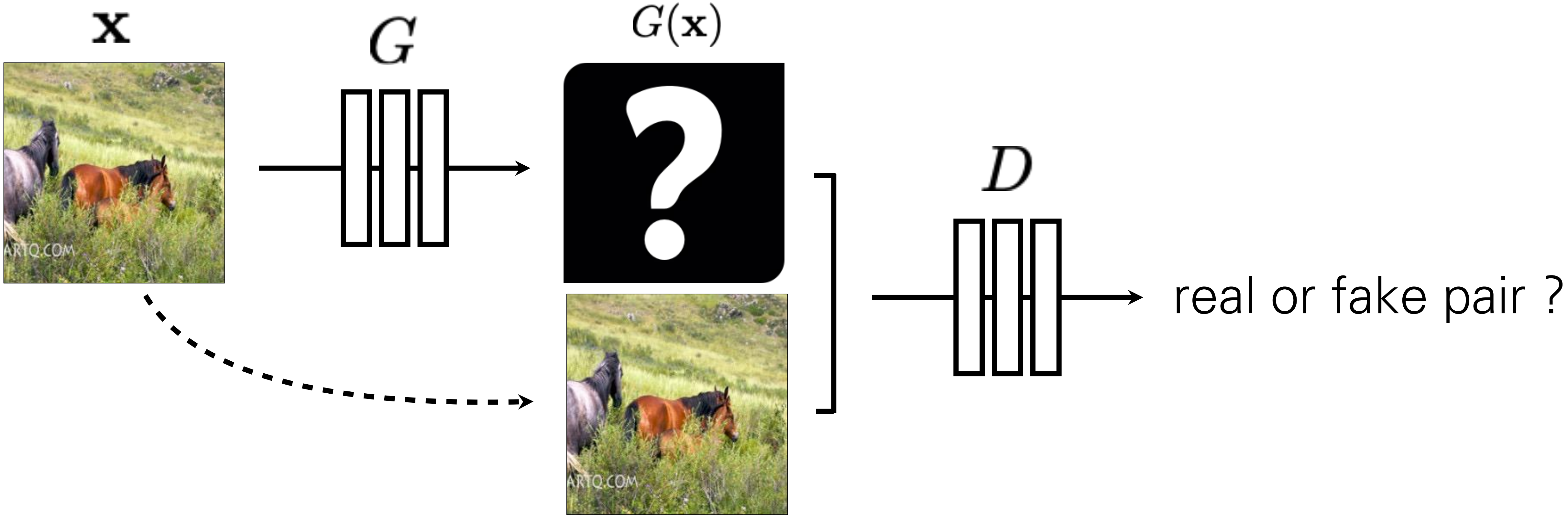


Cycle-Consistent Adversarial Networks



$$\arg\min_G \max_D \ \mathbb{E}_{\mathbf{x},\mathbf{y}}[\ \log D(\mathbf{x}, G(\mathbf{x})) + \log(1 - D(\mathbf{x}, \mathbf{y}))\ ]$$

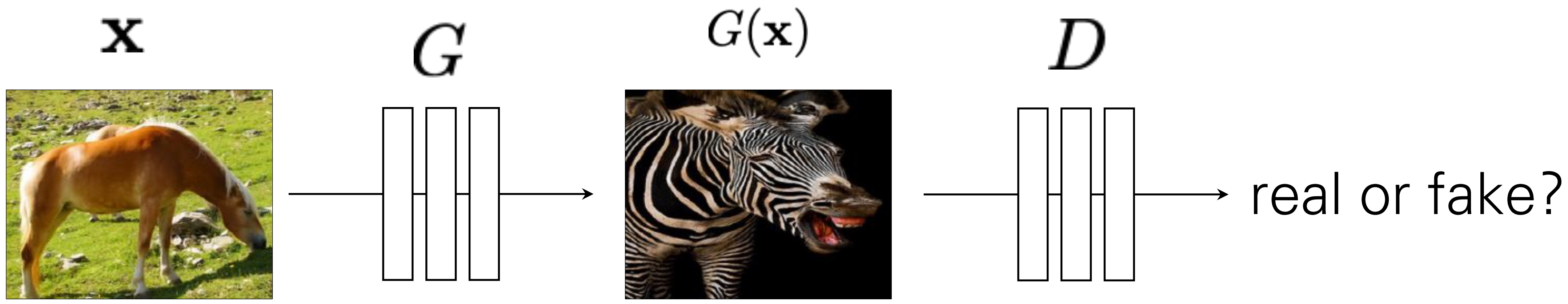
Cycle-Consistent Adversarial Networks



$$\arg\min_G \max_D \ \mathbb{E}_{\mathbf{x},\mathbf{y}}[\ \log D(\mathbf{x}, G(\mathbf{x})) + \log(1 - D(\mathbf{x},\mathbf{y}))\ ]$$

No input-output pairs!

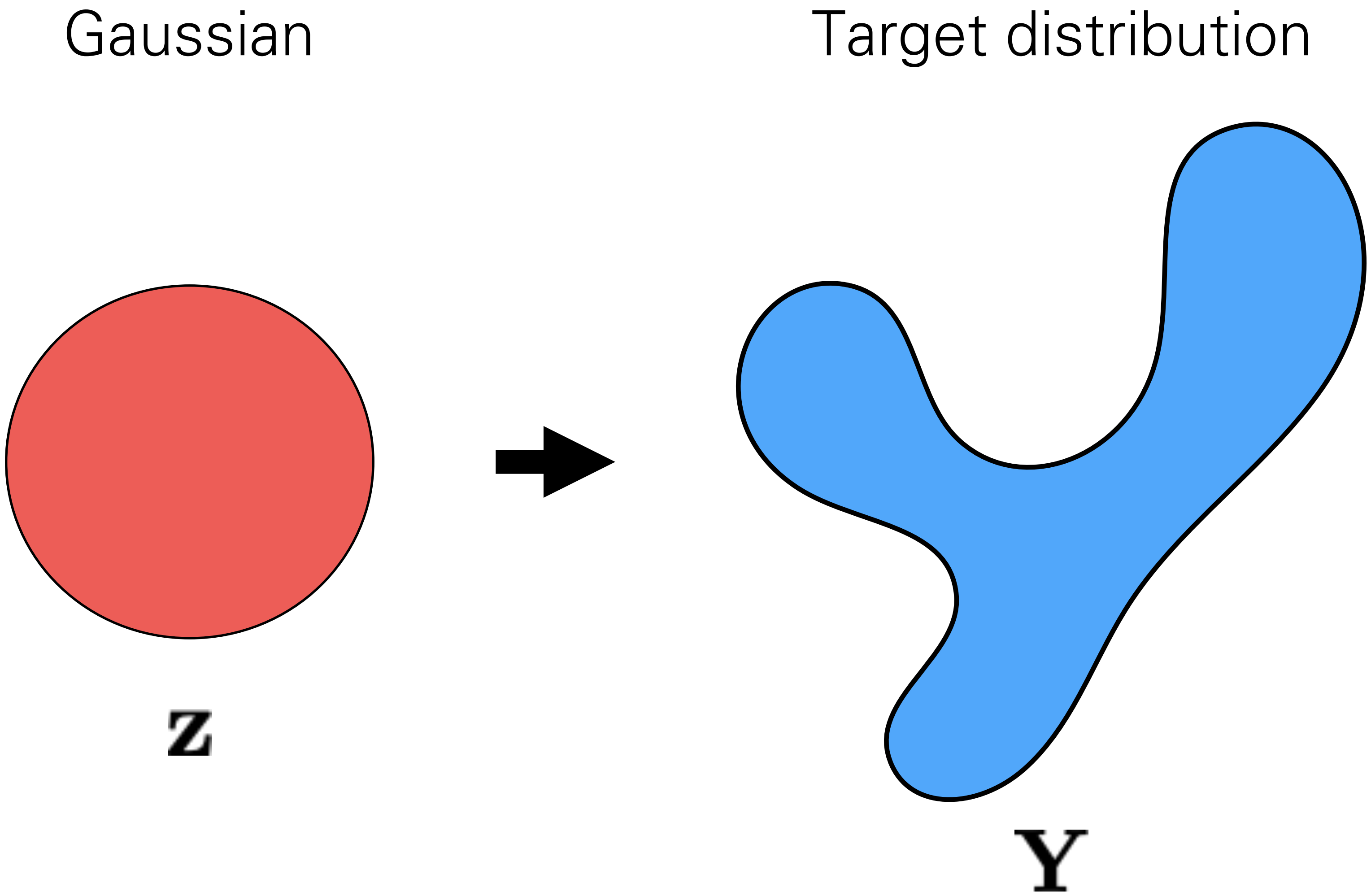
Cycle-Consistent Adversarial Networks



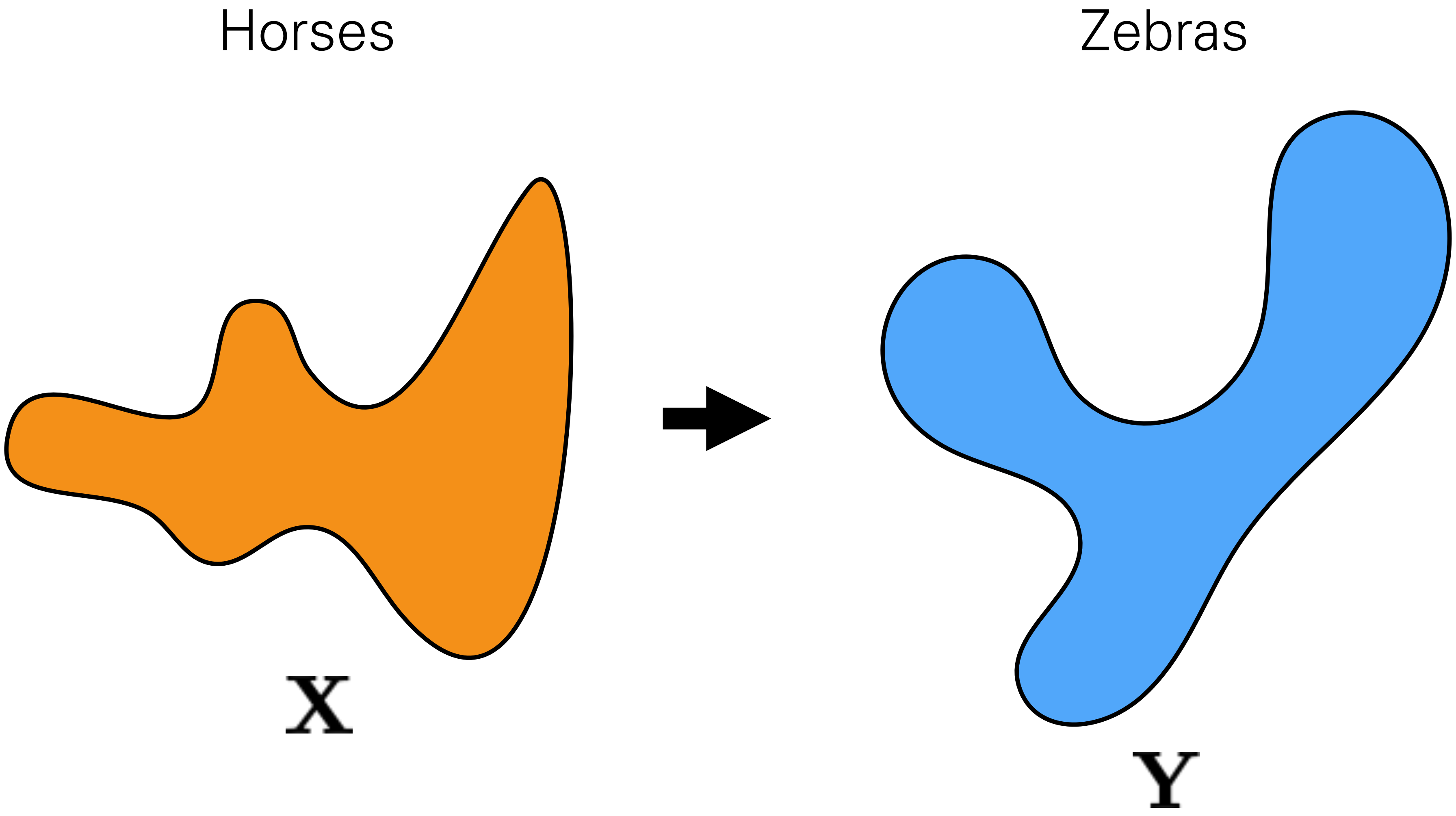
$$\arg\min_{G}\max_{D}\ \mathbb{E}_{\mathbf{x},\mathbf{y}}[\ \log D(G(\mathbf{x}))\ \ +\ \ \log(1-D(\mathbf{y}))\ ]$$

- Usually loss functions check if output matches a target instance
- GAN loss checks if output is part of an admissible set

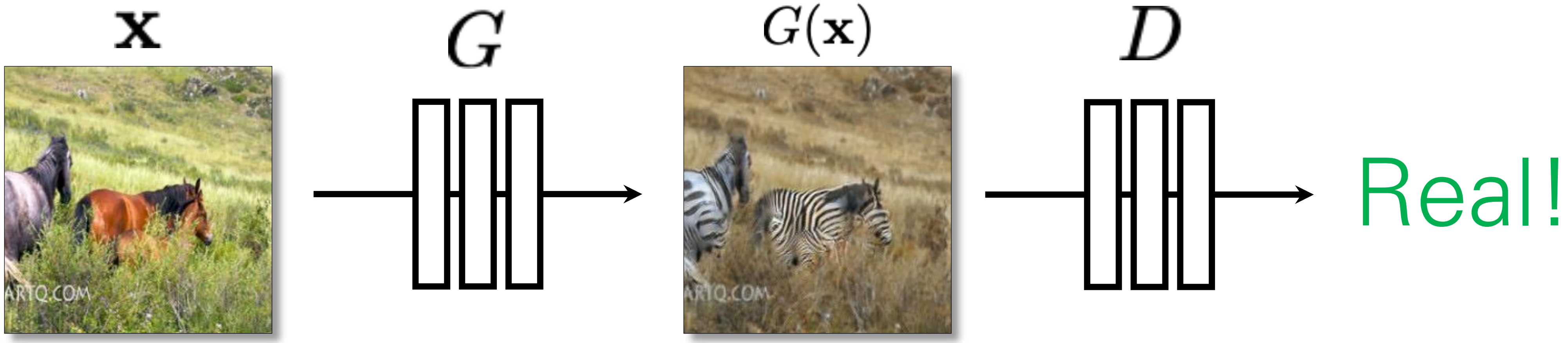
Cycle-Consistent Adversarial Networks



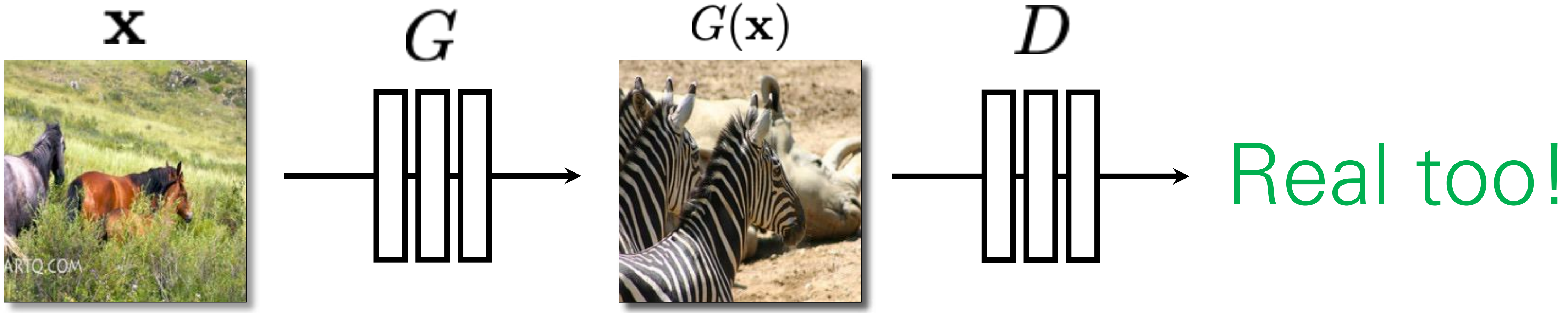
Cycle-Consistent Adversarial Networks



Cycle-Consistent Adversarial Networks

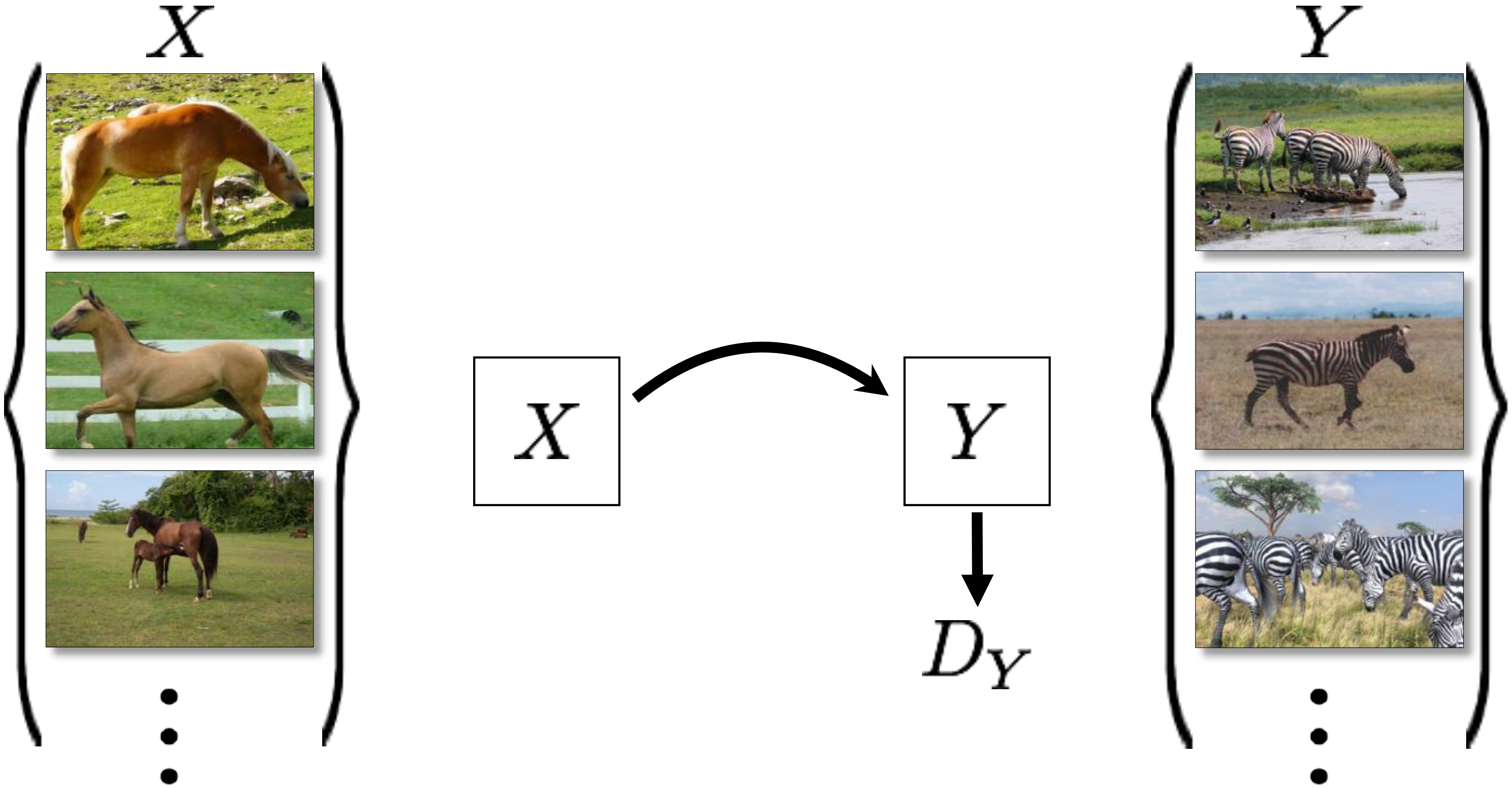


Cycle-Consistent Adversarial Networks



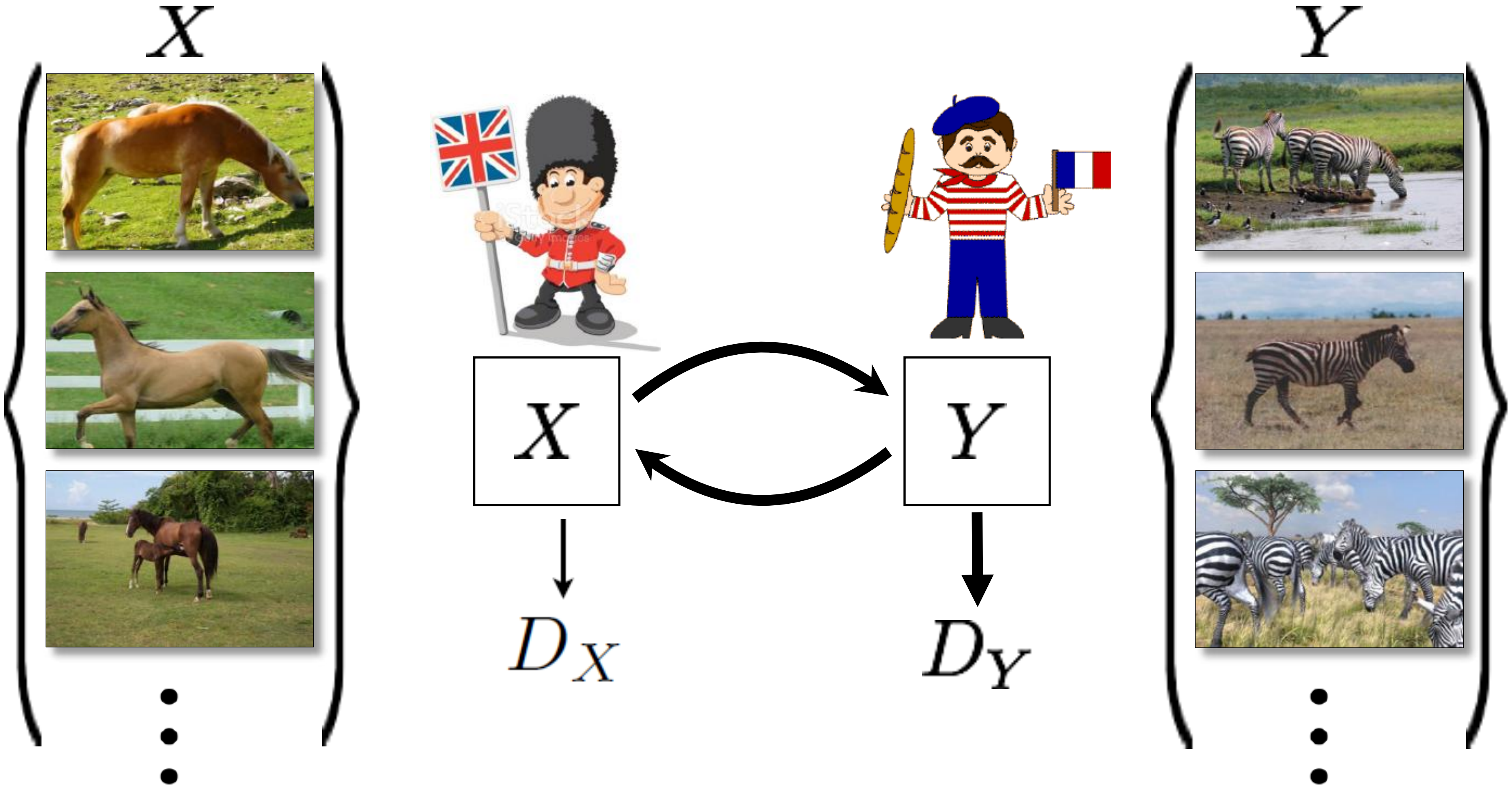
Nothing to force output to correspond to input

Cycle-Consistent Adversarial Networks

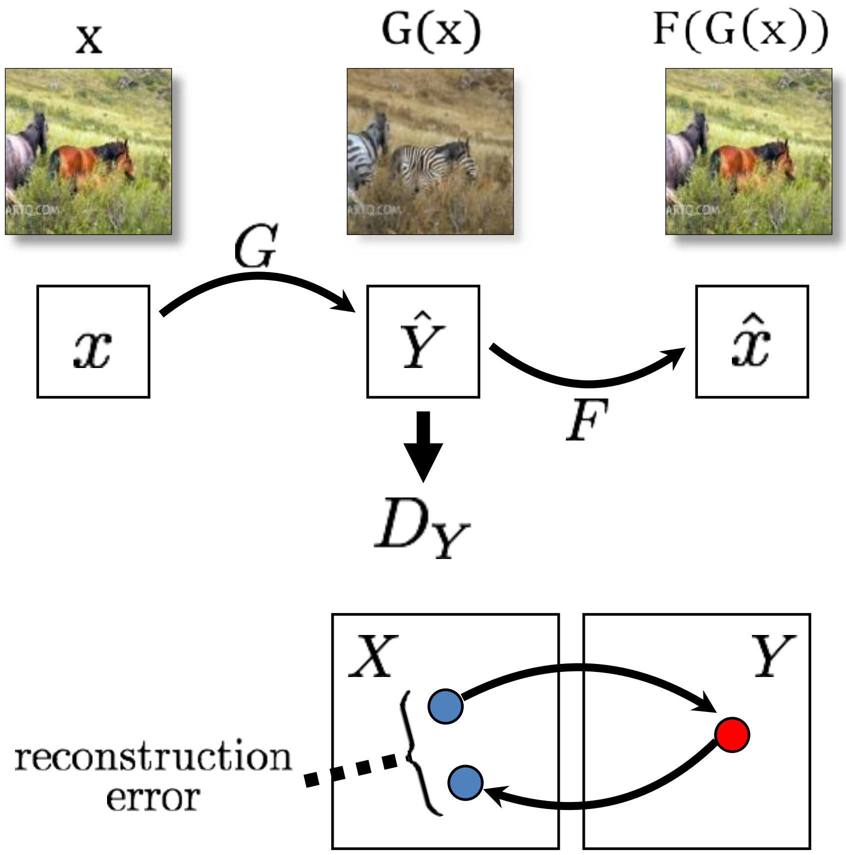


[Zhu et al. 2017], [Yi et al. 2017], [Kim et al. 2017]

Cycle-Consistent Adversarial Networks

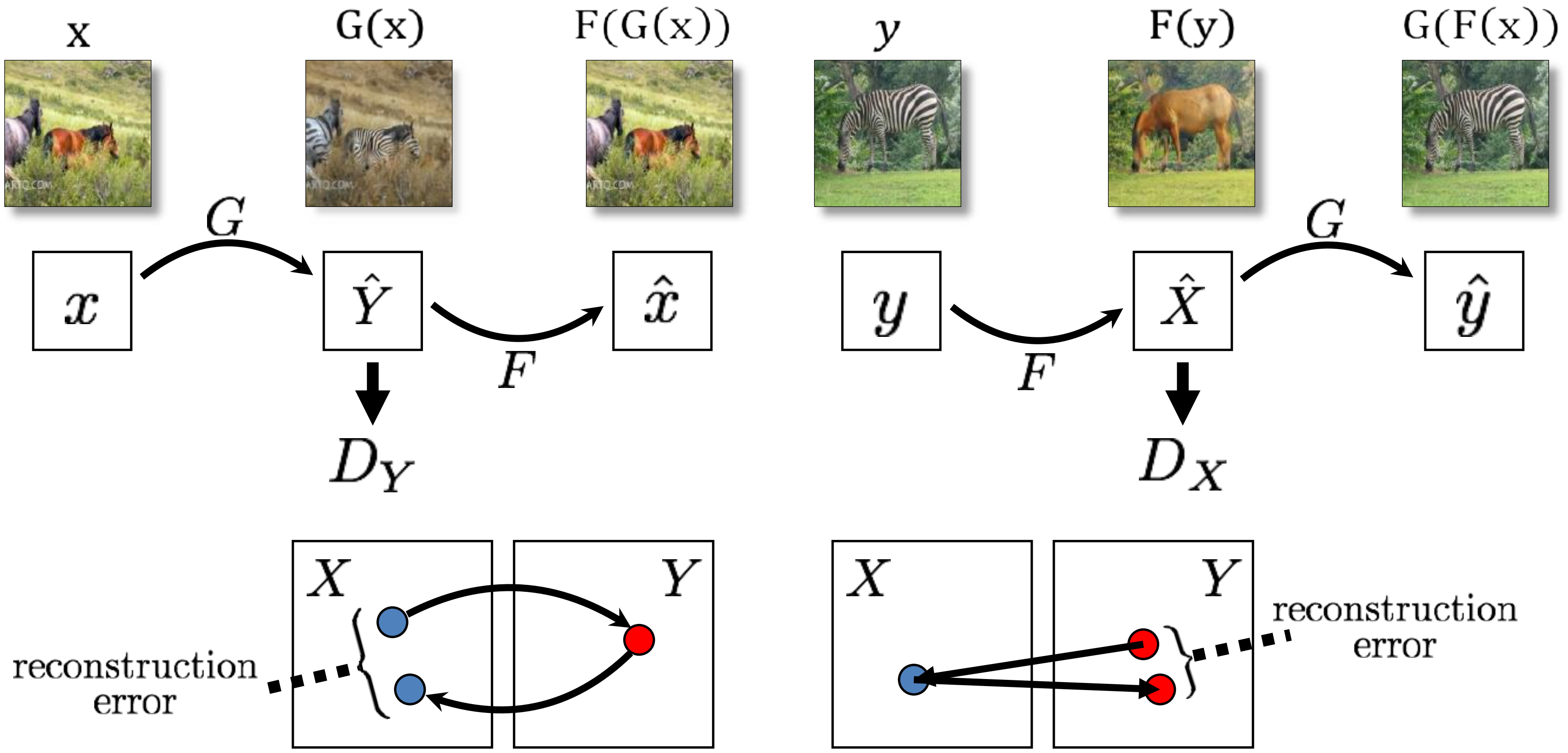


Cycle Consistency Loss



$$\left\|F(G(x)) - x\right\|_1$$

Cycle Consistency Loss



$$\left\| F(G(x)) - x \right\|_1 \qquad \left\| G(F(y)) - y \right\|_1$$

Cycle-Consistent Adversarial Networks

Cycle-Consistent Adversarial Networks

Cycle-Consistent Adversarial Networks

Photograph
@ Alexei Efros

Monet

Van Gogh

Cezanne

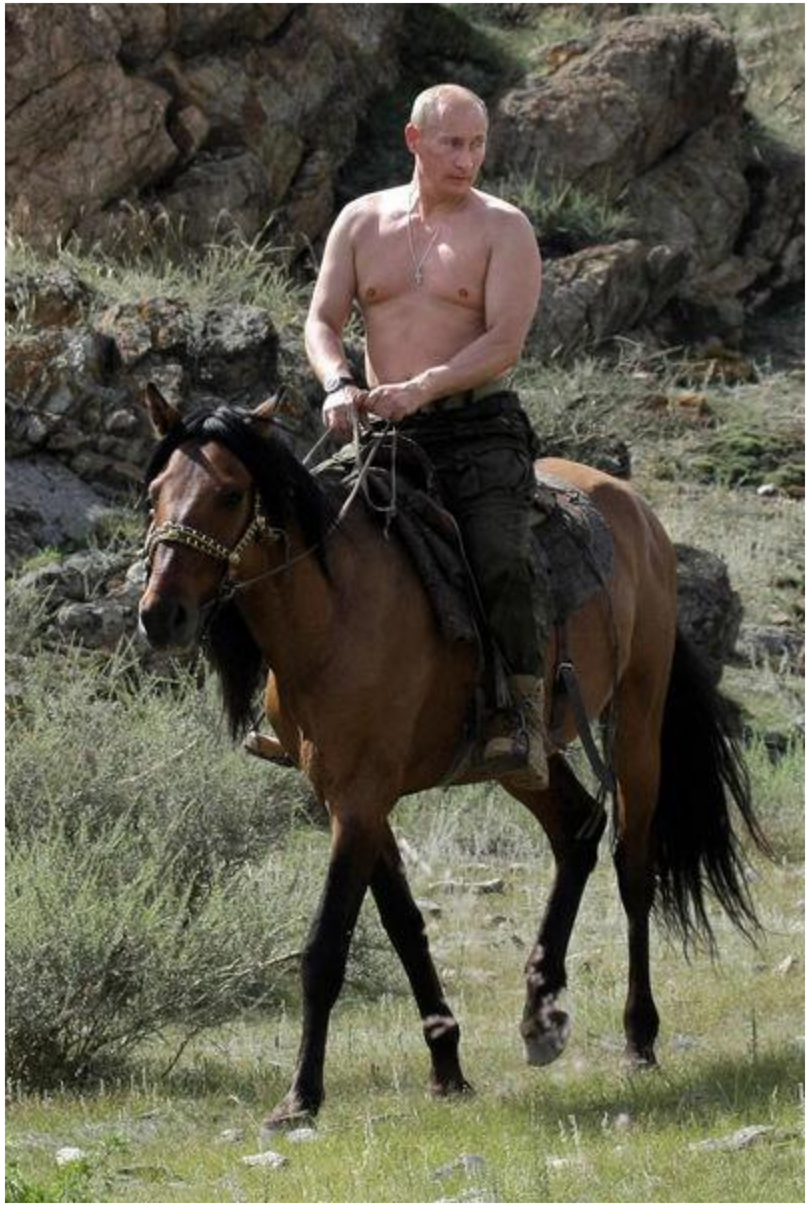
Ukiyo-e


Input
Monet
Van Gogh
Cezanne
Ukiyo-e

Cycle-Consistent Adversarial Networks

Cycle-Consistent Adversarial Networks

Cycle-Consistent Adversarial Networks

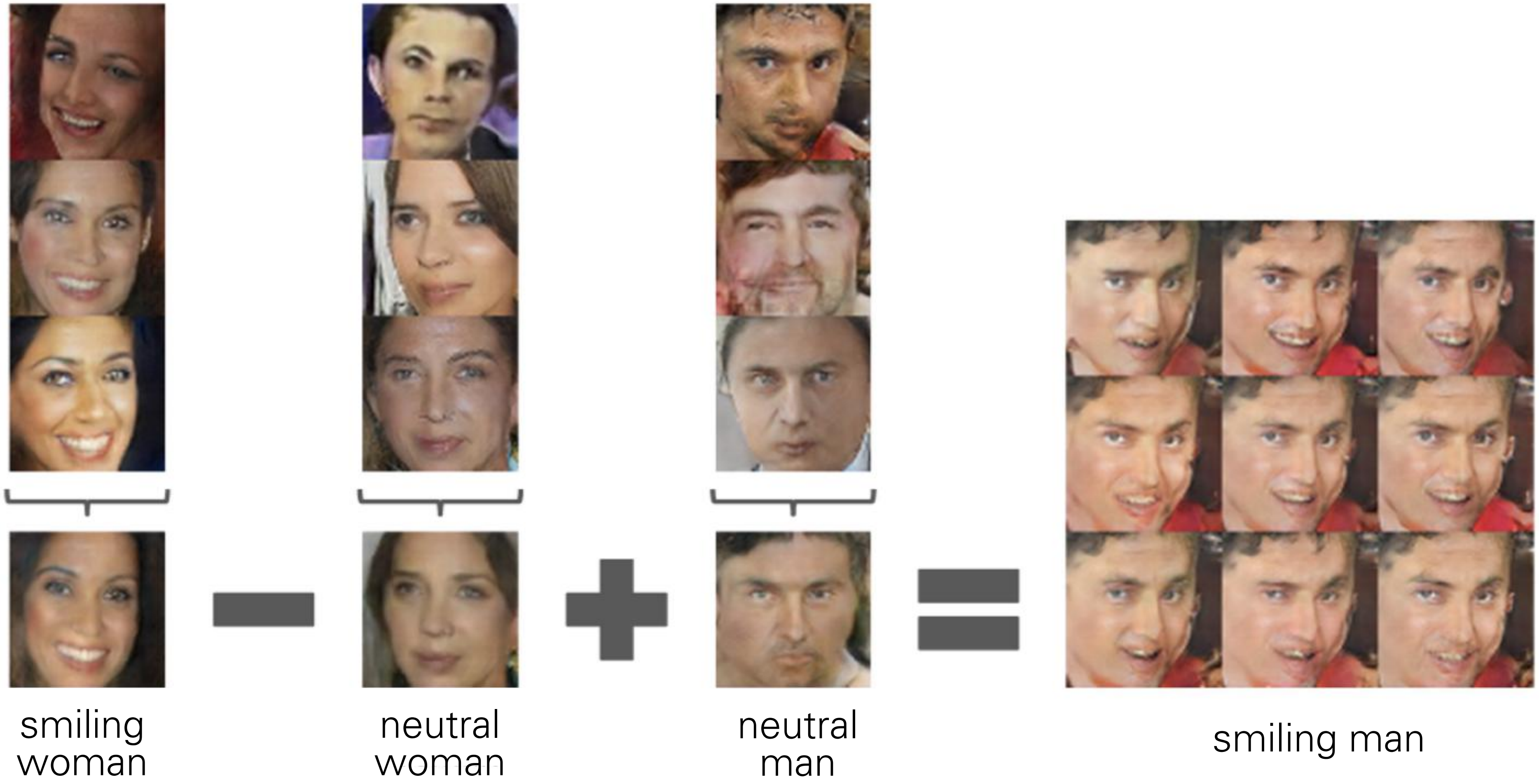


Cycle-Consistent Adversarial Networks

Lecture overview

- Motivation and Definition of Implicit Models
- Original GAN (Goodfellow et al, 2014)
- Evaluation: Parzen, Inception, Frechet
- Theory of GANs
- GAN Progression
- Conditional GANs, Cycle-Consistent Adversarial Networks
- **GANs and Representations**
- Applications

DCGAN Revisited: Vector Arithmetic

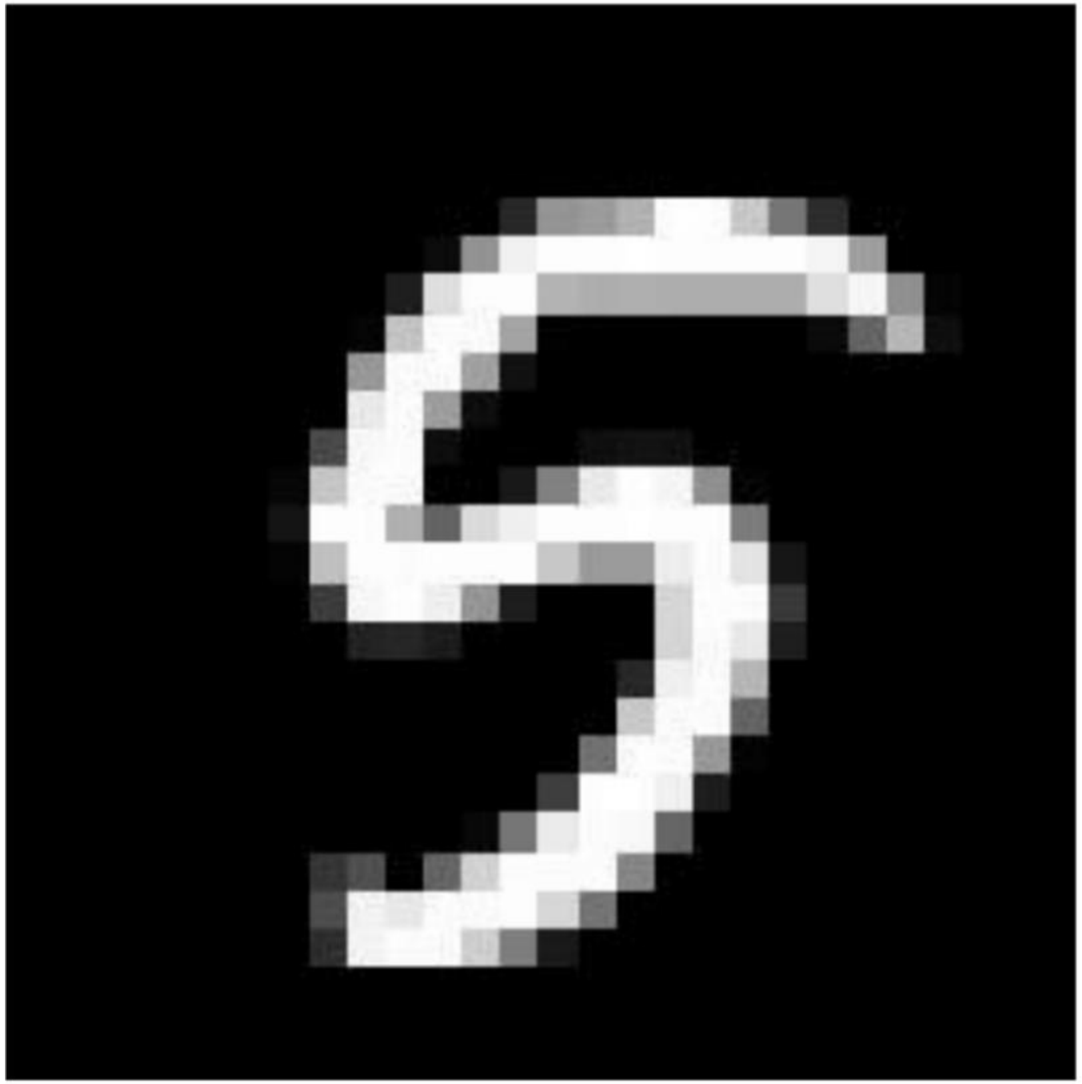


[Radford et al 2016]

GANs for unsupervised feature learning

- InfoGAN (Information Maximizing GAN)
- BiGAN (Bidirectional Generative Adversarial Networks)
  ALI (Adversarially Learned Inference)
  - BigBiGAN (Big Bidirectional Generative Adversarial Networks)

InfoGAN



```
array([[151, 157, 250, ...,  20,   0,   0],
       [148, 161, 242, ...,  15,   0,   0],
       [235, 228, 255, ...,   3,   0,   0],
       ...,
       [252, 254, 176, ..., 240, 253, 253],
       [253, 253, 253, ..., 253, 200, 200],
       [253, 253, 253, ..., 253, 200, 200]]
, dtype=uint8)
```

InfoGAN

```
array([[151, 157, 250, ...,  20,   0,   0],
       [148, 161, 242, ...,  15,   0,   0],
       [235, 228, 255, ...,   3,   0,   0],
       ...,
       [252, 254, 176, ..., 240, 253, 253],
       [253, 253, 253, ..., 253, 200, 200],
       [253, 253, 253, ..., 253, 200, 200]]
, dtype=uint8)
```

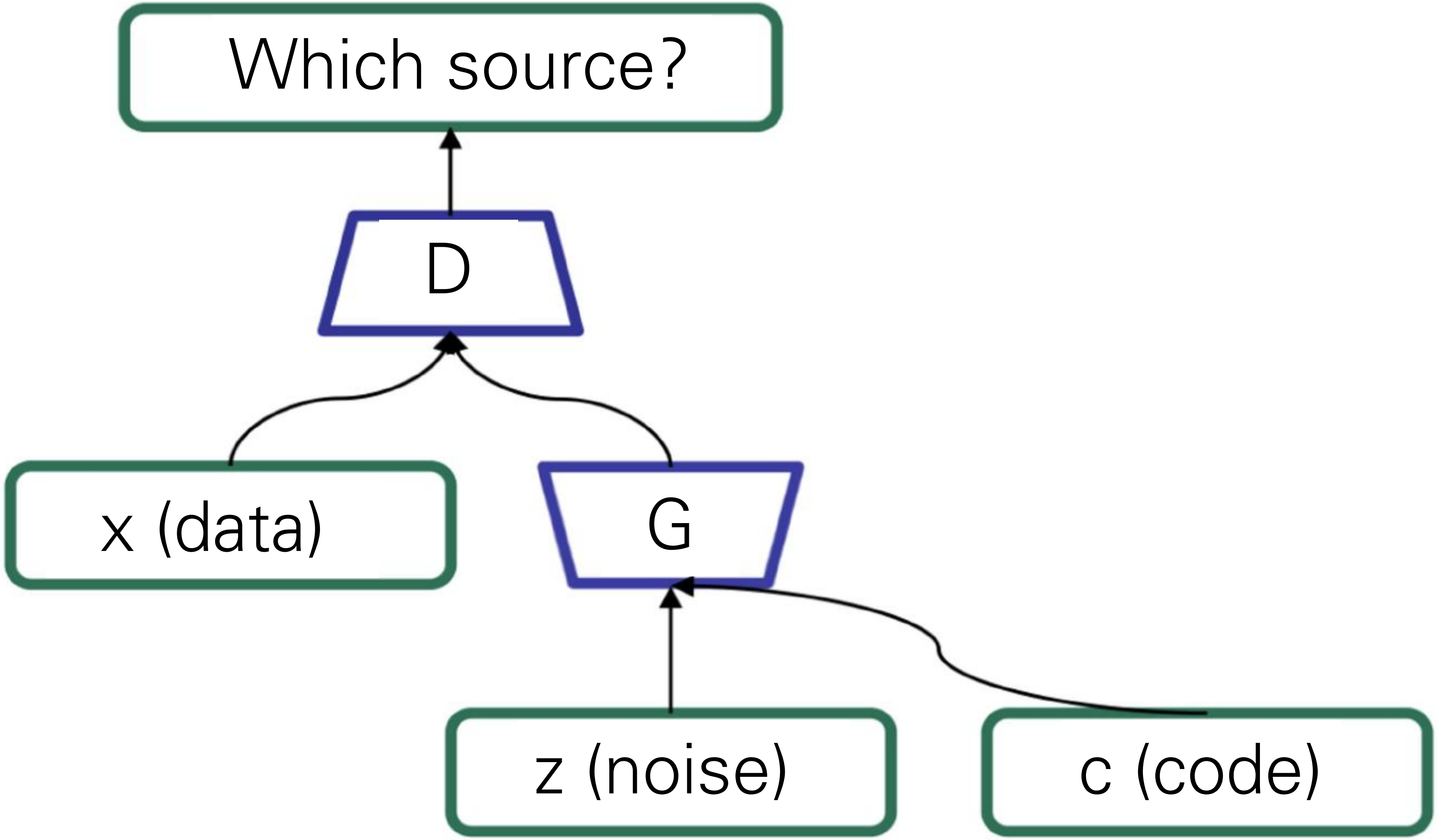
Simple factors interact to create complex observations.

Digit type: “5”
Rotation: Tilting to the right
Width: Medium
……

Data: x

Latent code: c

InfoGAN



InfoGAN

- Simple idea: Independent factors in latent code should maximally explain variations in generated images

- Formally: We want to maximize the mutual information between latent code and generated images:

$$\max_G I(c;x) = H(x) - H(x|c)$$
$$= H(c) - H(c|x)$$

where $x = G(z,c)$

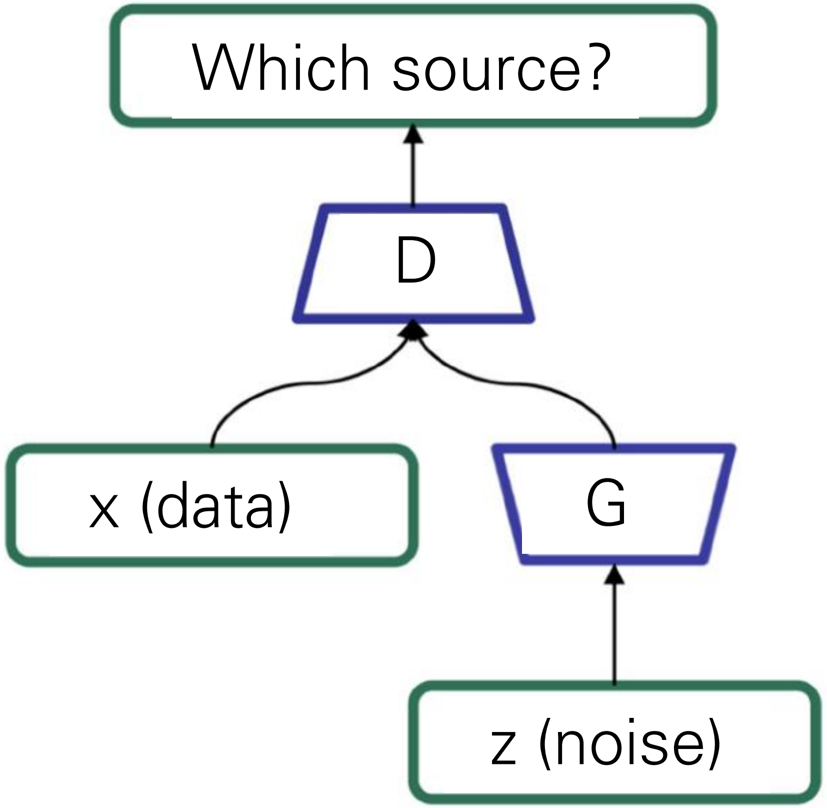
InfoGAN

- Mutual information can be maximized easily with a variational lower bound:

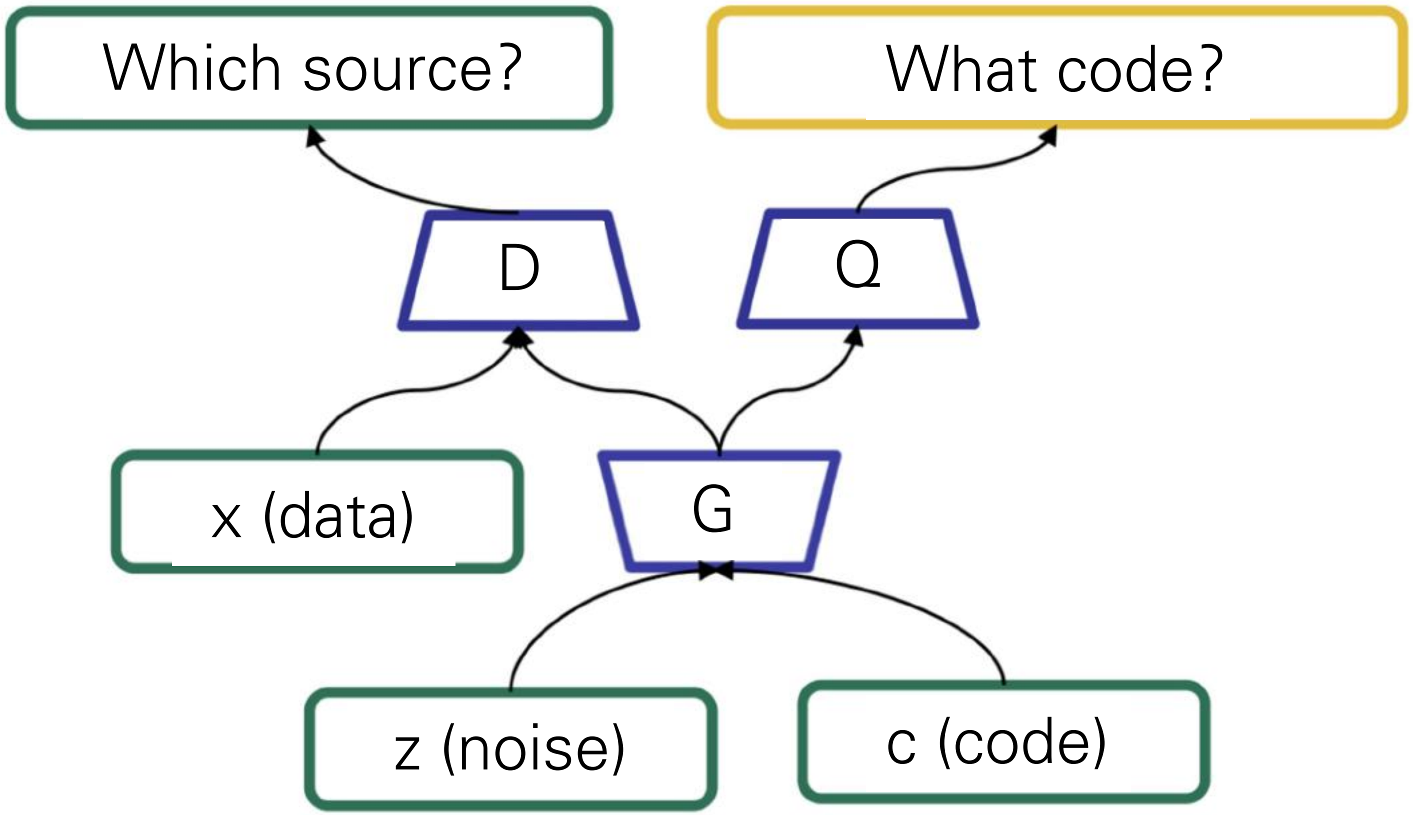
$$
\begin{aligned}
I(c; G(z, c)) &= H(c) - H(c|G(z, c)) \\
&= H(c) + \mathbb{E}_{x \sim G(z,c)}[\mathbb{E}_{c' \sim P(c|x)}[\log P(c'|x)]] \\
&= H(c) + \mathbb{E}_{x \sim G(z,c)}[\underbrace{D_{\mathrm{KL}}(P(\cdot|x) \parallel Q(\cdot|x))}_{\geq 0} + \mathbb{E}_{c' \sim P(c|x)}[\log Q(c'|x)]] \\
&\geq H(c) + \mathbb{E}_{x \sim G(z,c)}[\mathbb{E}_{c' \sim P(c|x)}[\log Q(c'|x)]]
\end{aligned}
$$

Simply MLE for a classifier/regressor

InfoGAN

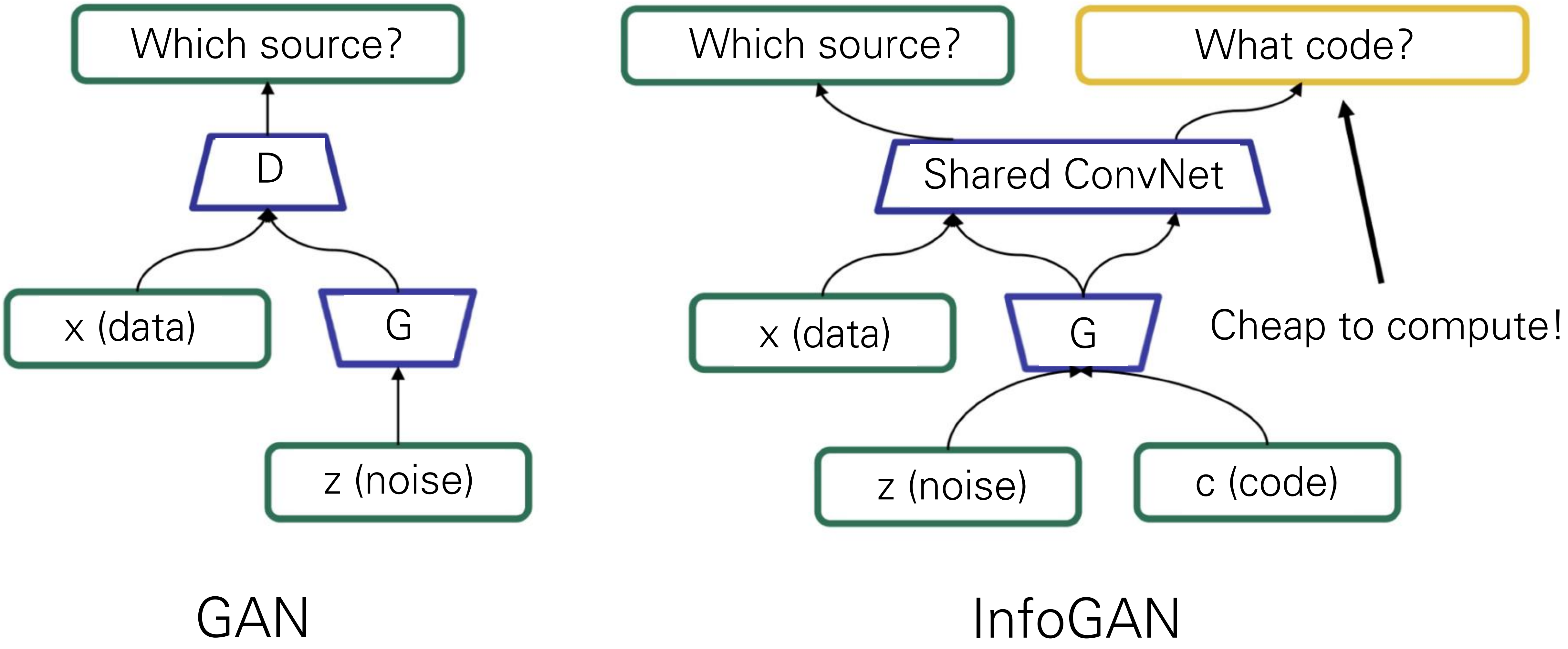


GAN

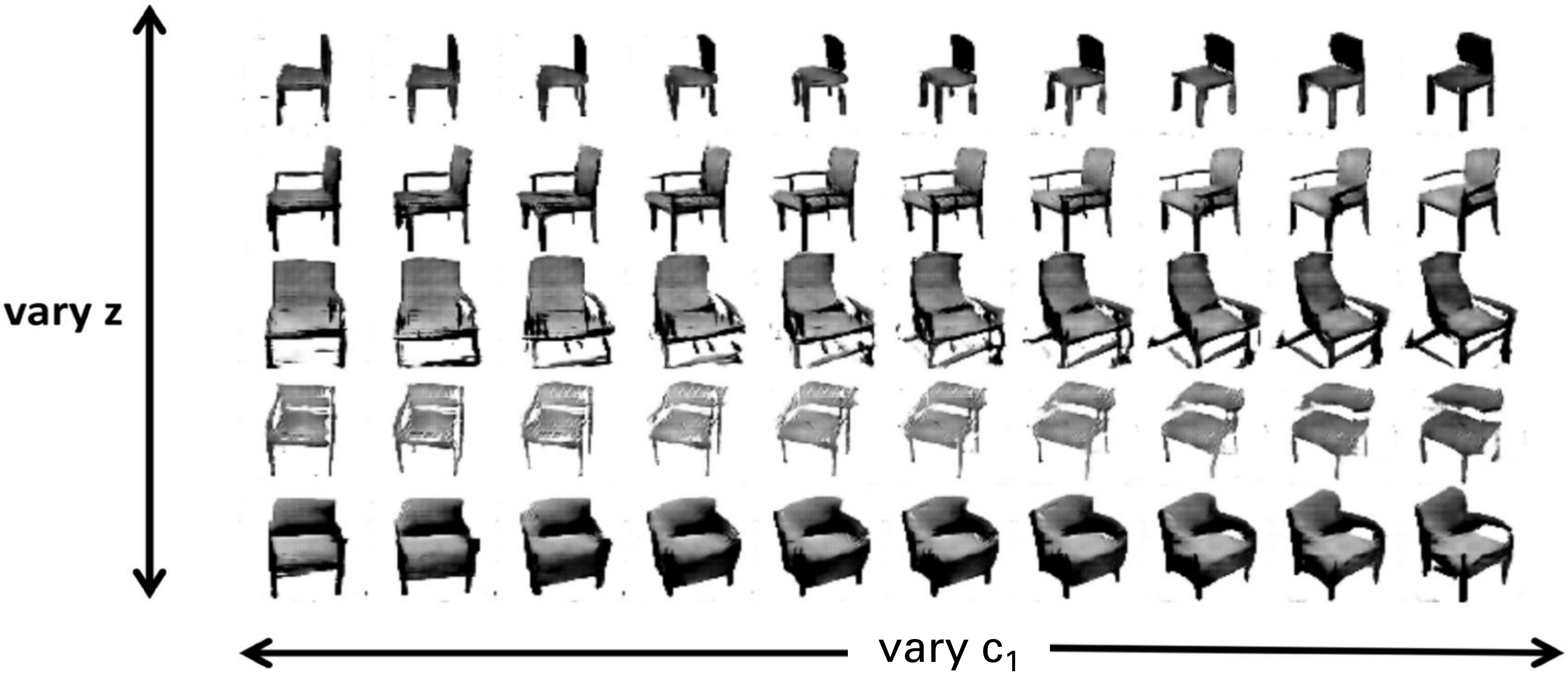


InfoGAN

InfoGAN

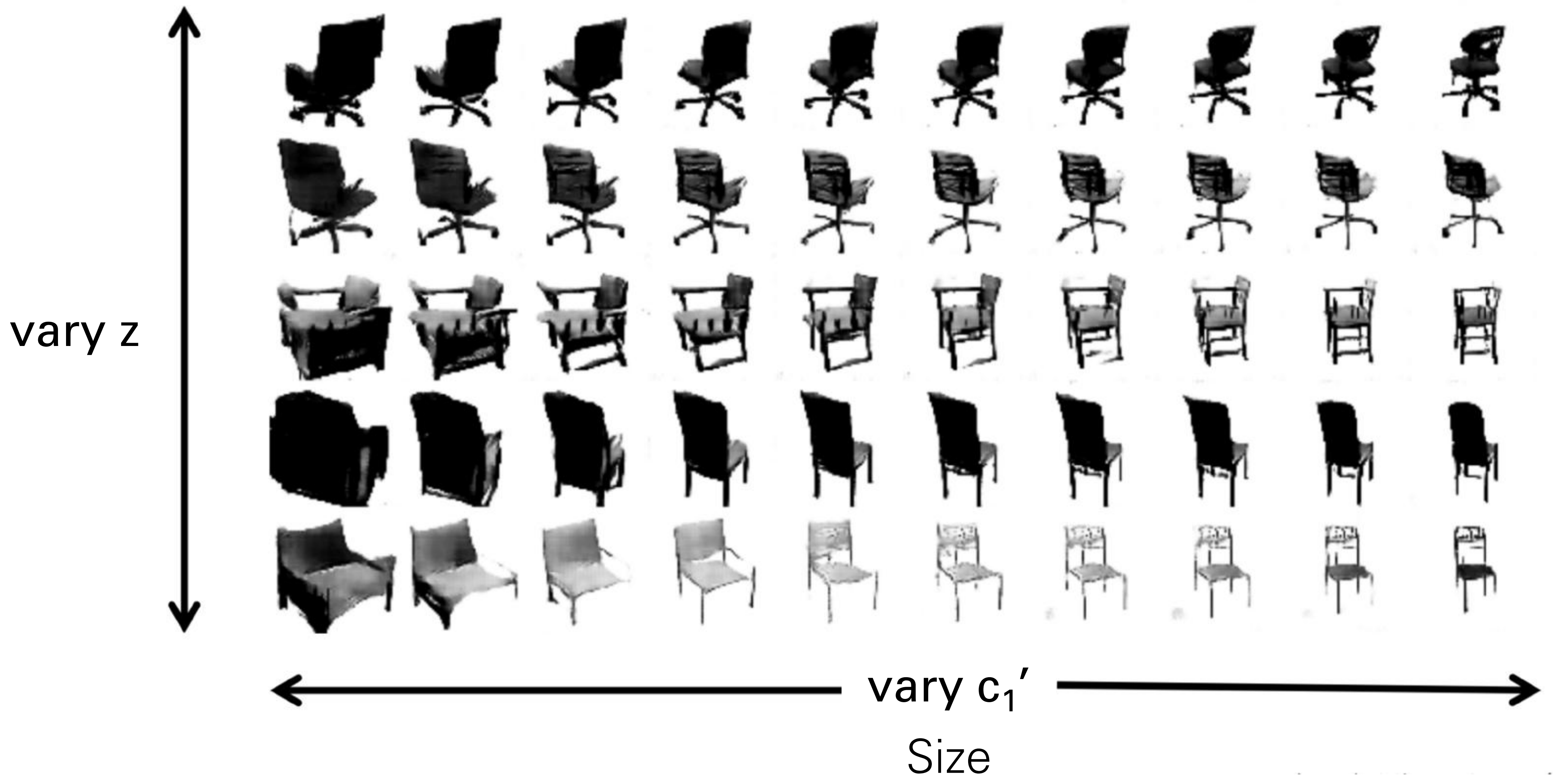

Which source?
D
x (data)
G
z (noise)
GAN
Which source?
What code?
Shared ConvNet
x (data)
G
Cheap to compute!
z (noise)
c (code)
InfoGAN

InfoGAN

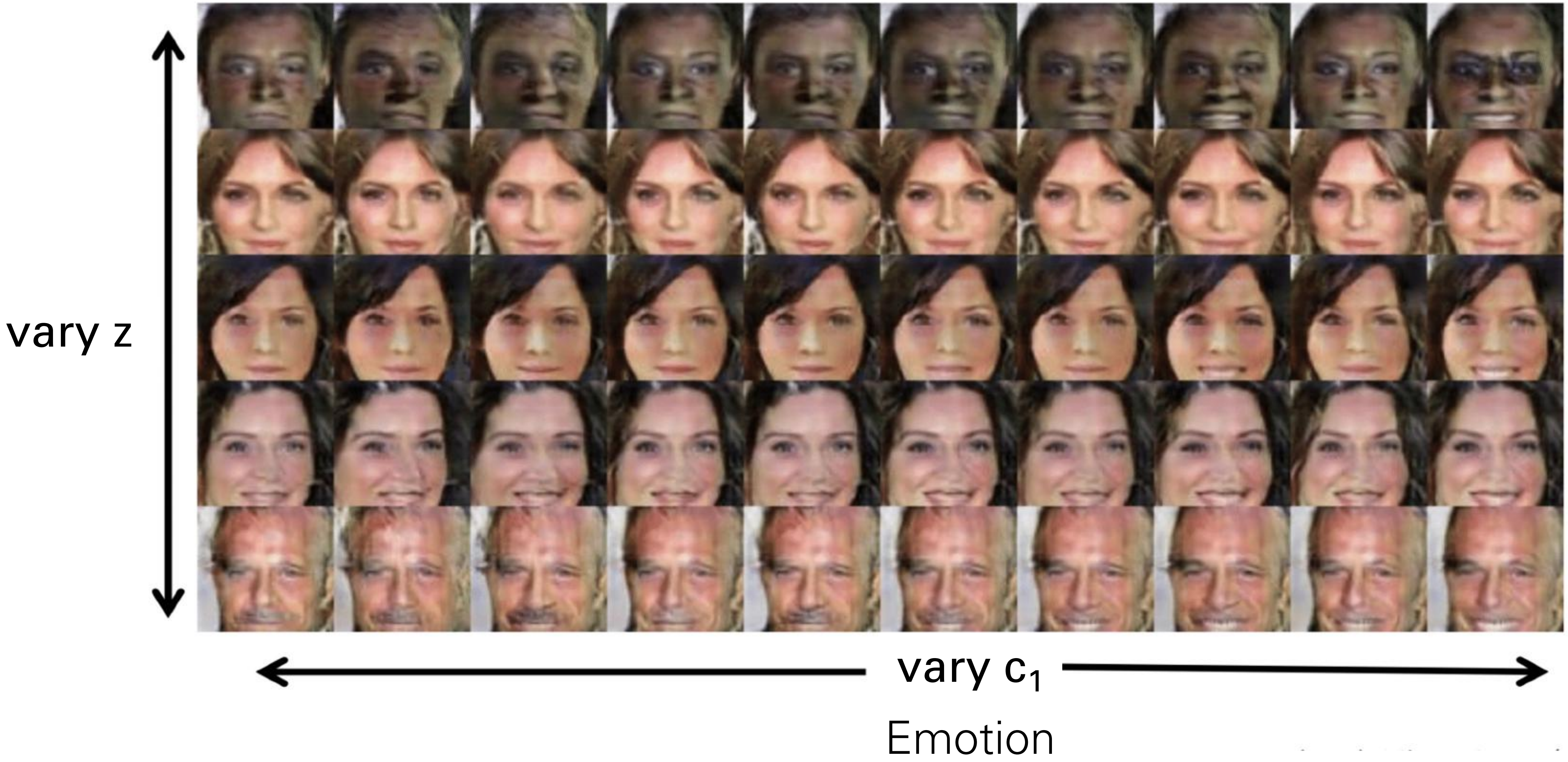


Rotation

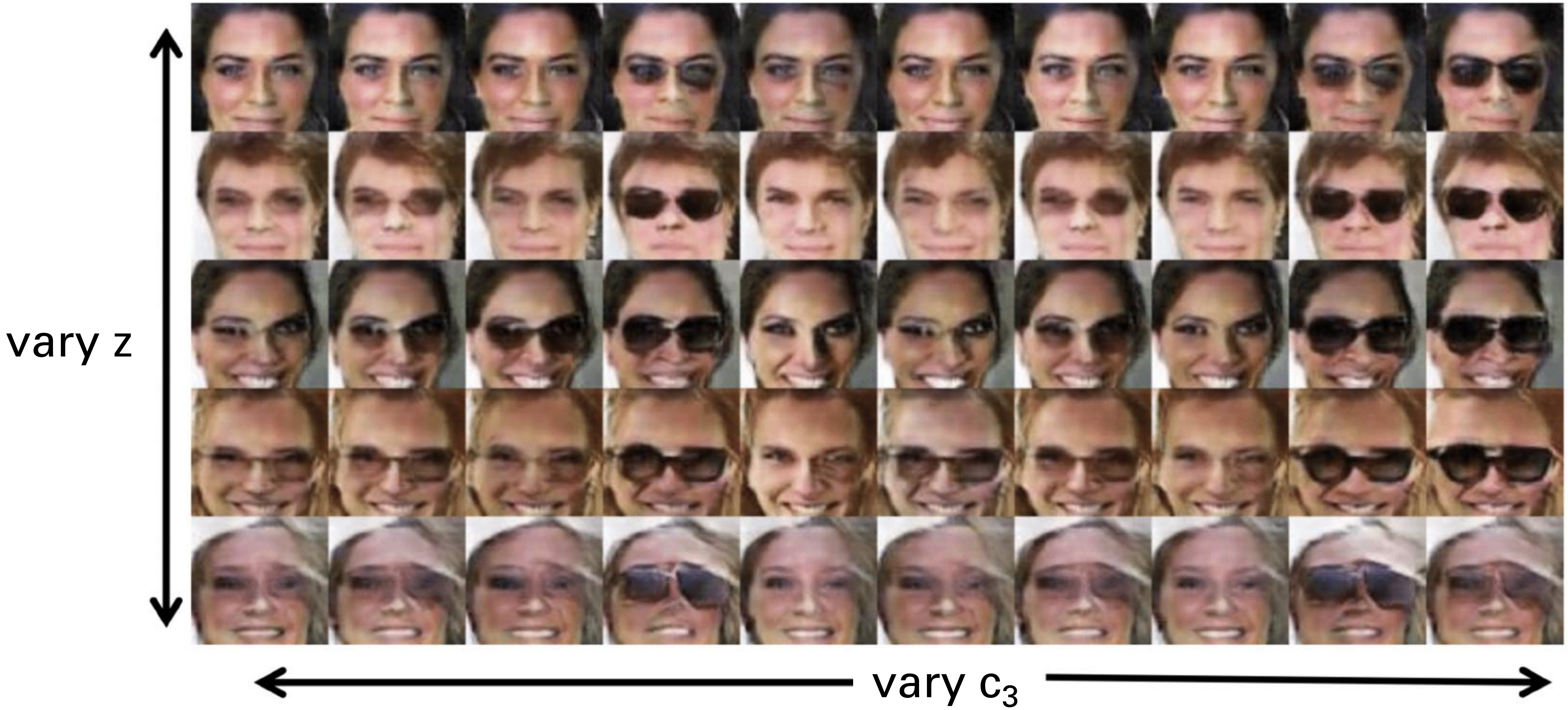
InfoGAN



InfoGAN



InfoGAN



Presence/absence of glasses

Unsupervised Category Discovery - BigGAN

- Trained with no labels!

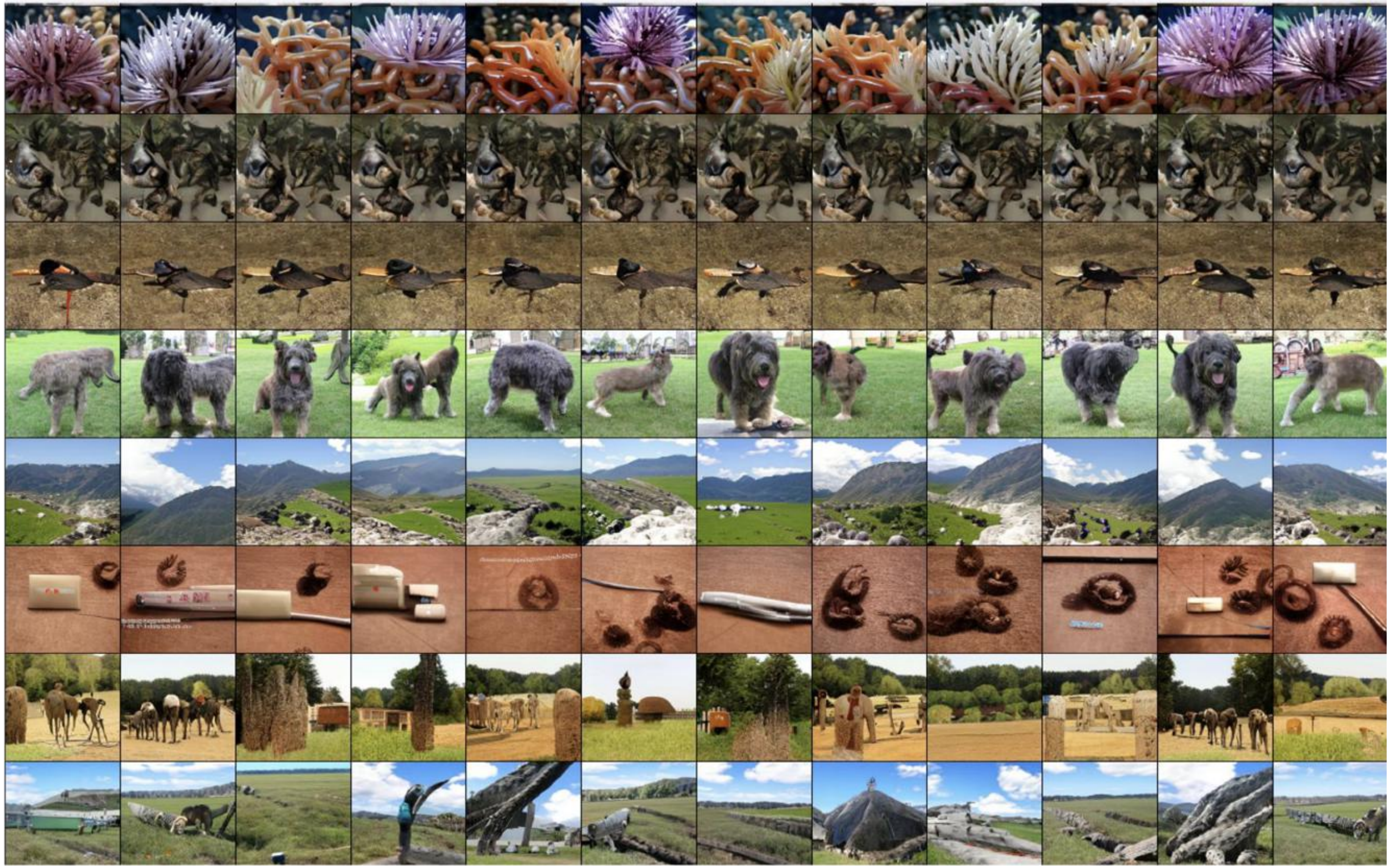
  z = concat([

  (a) $[N(0,I)]^{120}$,

  (b) UniformCateg(1024)

  ])

- Each row is one value of the categorical (b); columns are Gaussian samples (a)



Slide credit: Jeff Donahue

Unsupervised Category Discovery - BigGAN

- Trained with no labels!

  z = concat([

  (a) $[N(0,I)]^{120}$,

  (b) UniformCateg(1024)

  ])

- Each row is one value of the categorical (b); columns are Gaussian samples (a)

Slide credit: Jeff Donahue

But what about inference…

- How can we use generative models?
  - GANs can generate content, but somethings you want to make inference about observed data.

- Can we incorporate an inference mechanism into GANs?

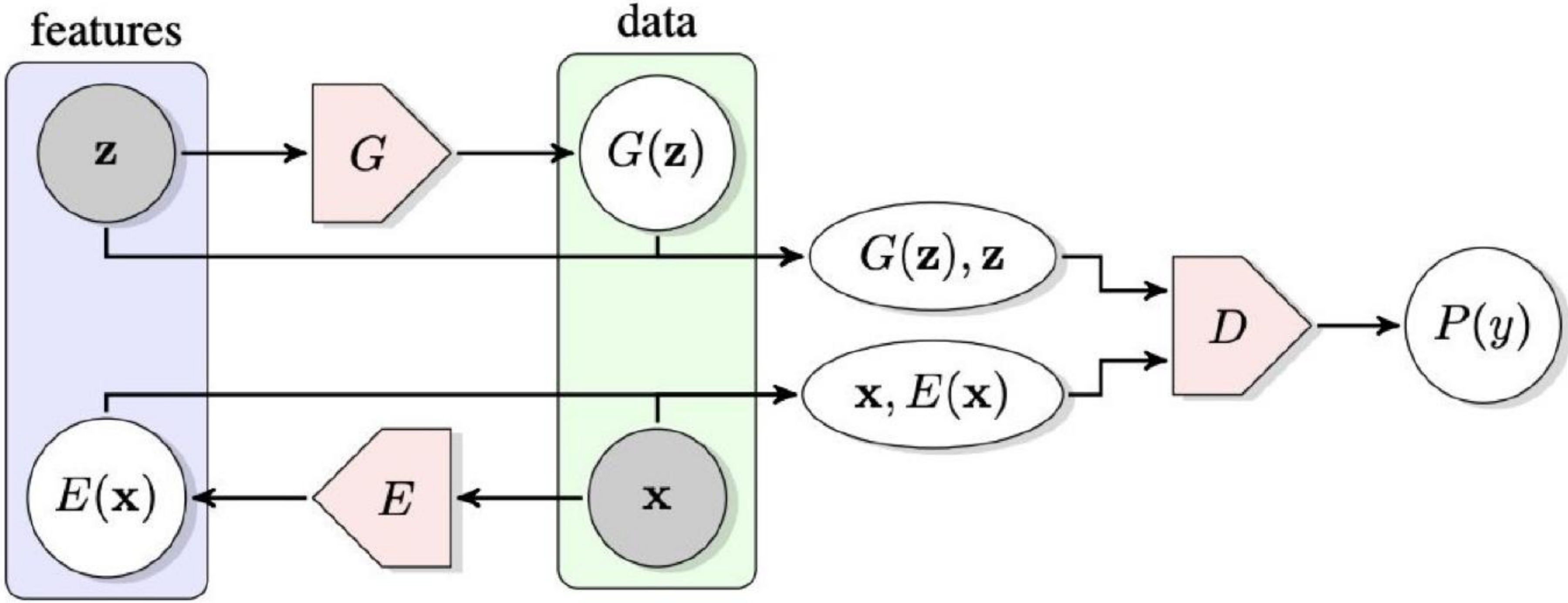
- Can we learn an inference mechanisms using an adversarial training paradigm?

Two papers, one model

- **ALI:** Vincent Dumoulin, Ishmael Belghazi, Olivier Mastropietro
  ADVERSARIALLY LEARNED INFERENCE, ICLR 2017
  Ben Poole, Alex Lamb, Martin Arjovsky

- **BiGAN:** Donahue, Krähenbühl and Darrell (2016), ADVERSARIAL FEATURE LEARNING, ICLR 2017

Adversarially Learned Inference (ALI)



- **Idea:** Cast the learning of both an inference model (encoder) and a generative model (decoder) in a GAN-like adversarial framework
- Discriminator is trained to discriminate between joint samples ($\mathbf{x}$, $\mathbf{z}$) from:
  - Encoder distribution $q(\mathbf{x}, \mathbf{z}) = q(\mathbf{x})\ \ q(\mathbf{z} \mid \mathbf{x})$, or
  - Decoder distribution $p(\mathbf{x}, \mathbf{z}) = p(\mathbf{z})\ \ p(\mathbf{x} \mid \mathbf{z})$.
- Generator learns conditionals $q(\mathbf{z} \mid \mathbf{x})$ and $p(\mathbf{x} \mid \mathbf{z})$ to fool the discriminator.

Adversarially Learned Inference (ALI)

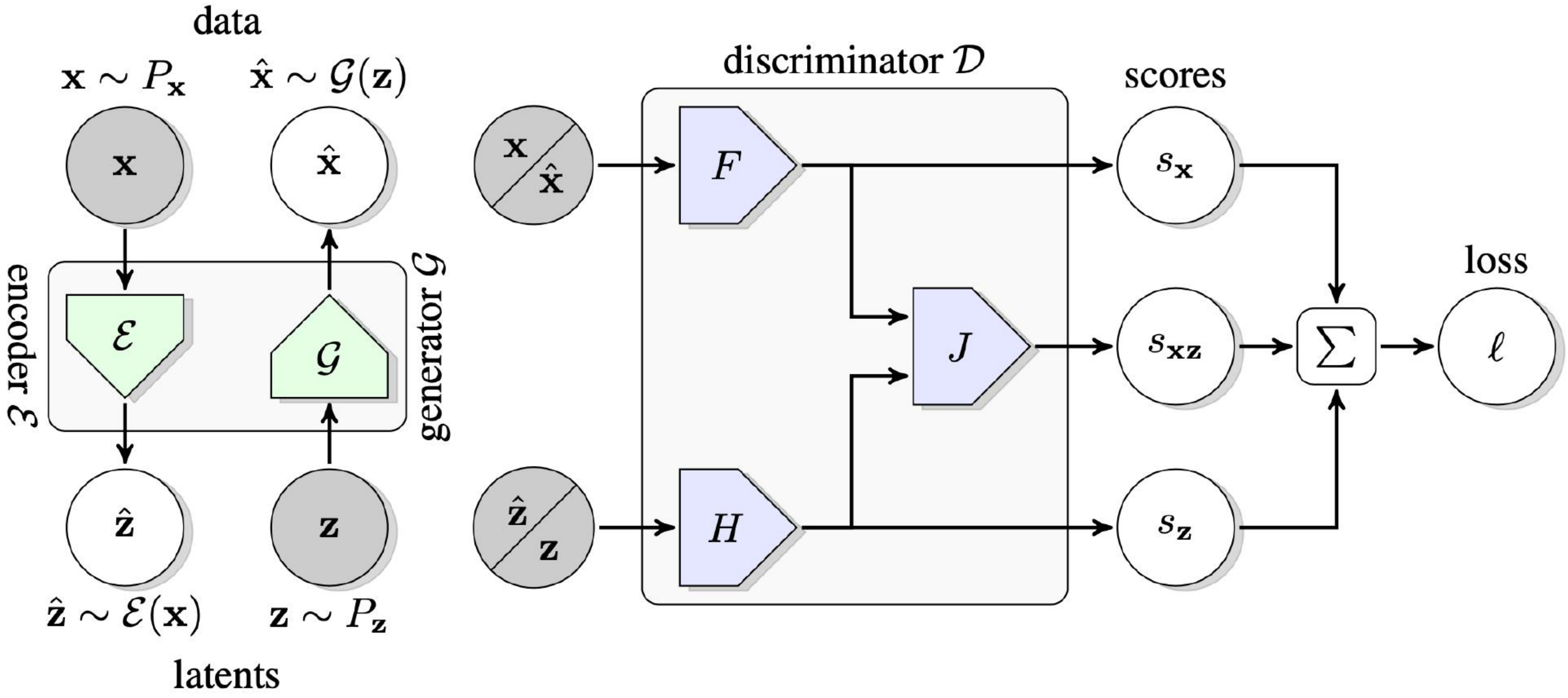
- In the global optimum, E and G are inverses; for all x and z we have
  - x = G(E(x))
  - z = E(G(z))
- In practice, this inversion property does not hold perfectly
  - Reconstructions still often capture interesting semantics

(a) CelebA samples.

(b) CelebA reconstructions.

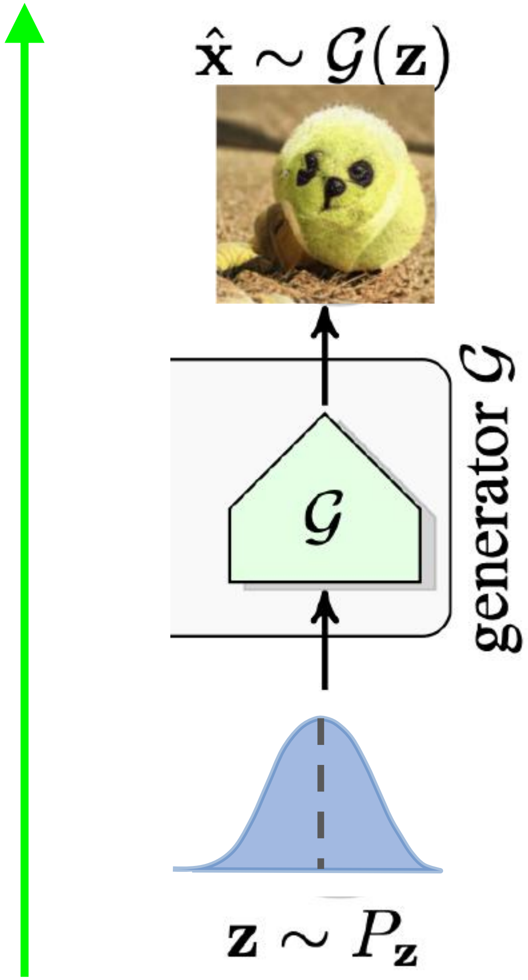
Slide credit: Jeff Donahue

Big Bidirectional GAN (BigBiGAN)



Slide credit: Jeff Donahue

BigBiGAN



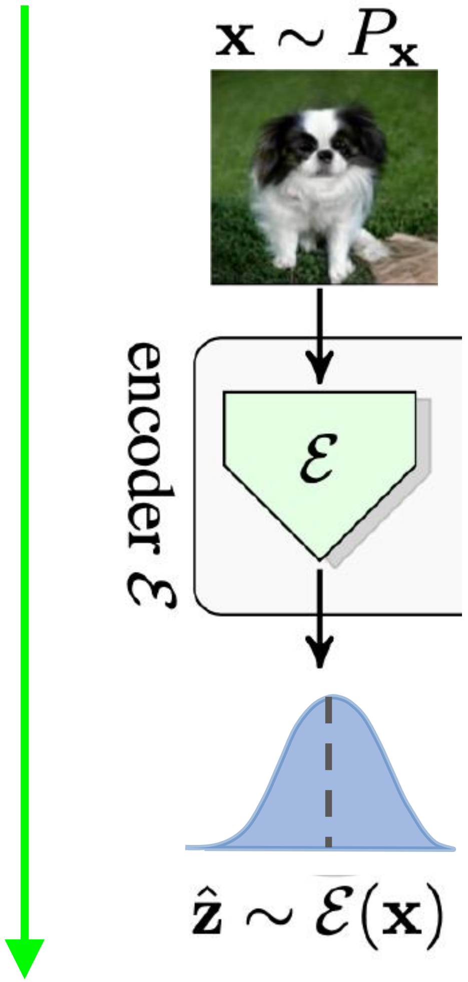
generated image

BigGAN generator convnet

latent sample

Slide credit: Jeff Donahue

BigBiGAN



real image sample

image recognition model (ResNet)

predicted latent

Slide credit: Jeff Donahue

BigBiGAN

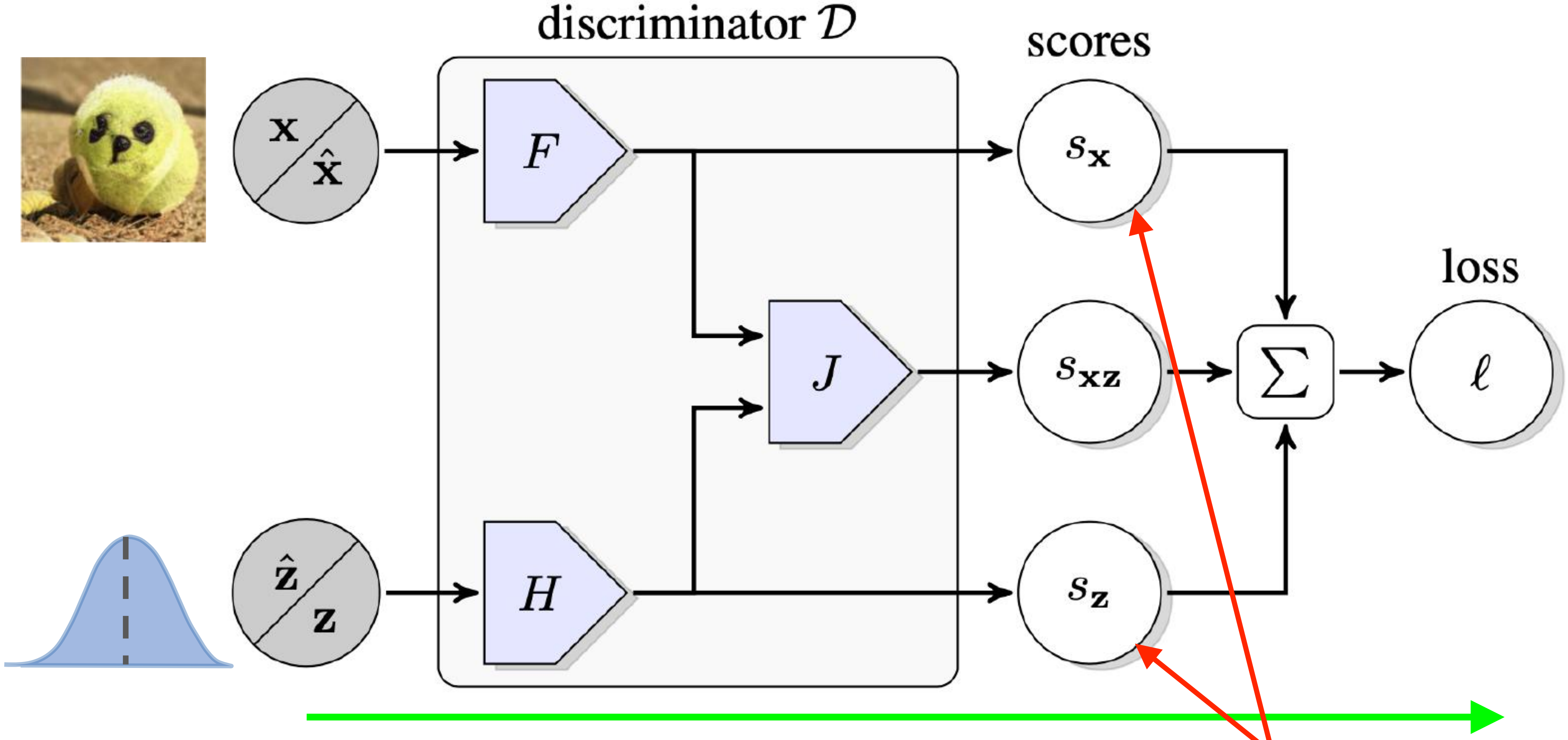
Discriminates between input pairs:

Encoder pair
(x, z' = E(x))

vs.

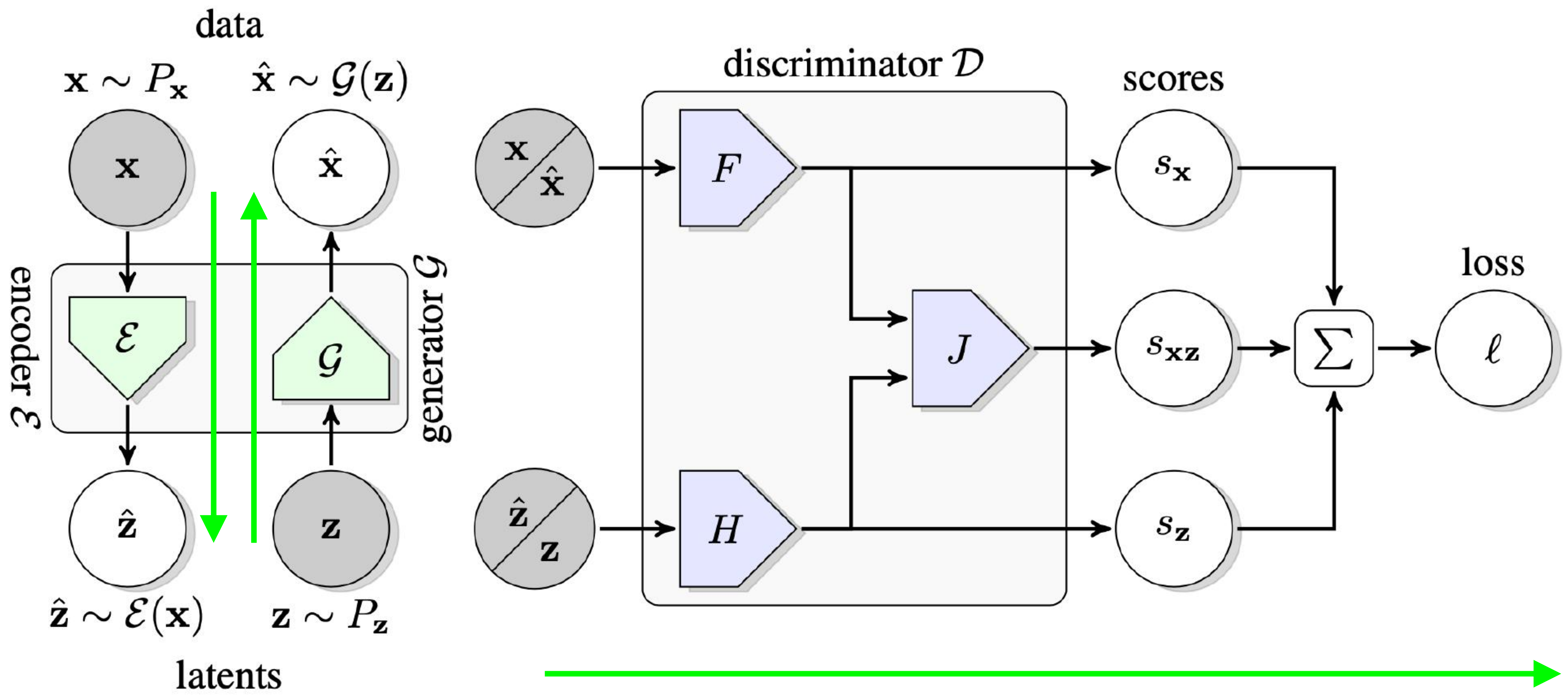
Generator pair
(x' = G(z), z)

sees images x and latents z (not just images x)



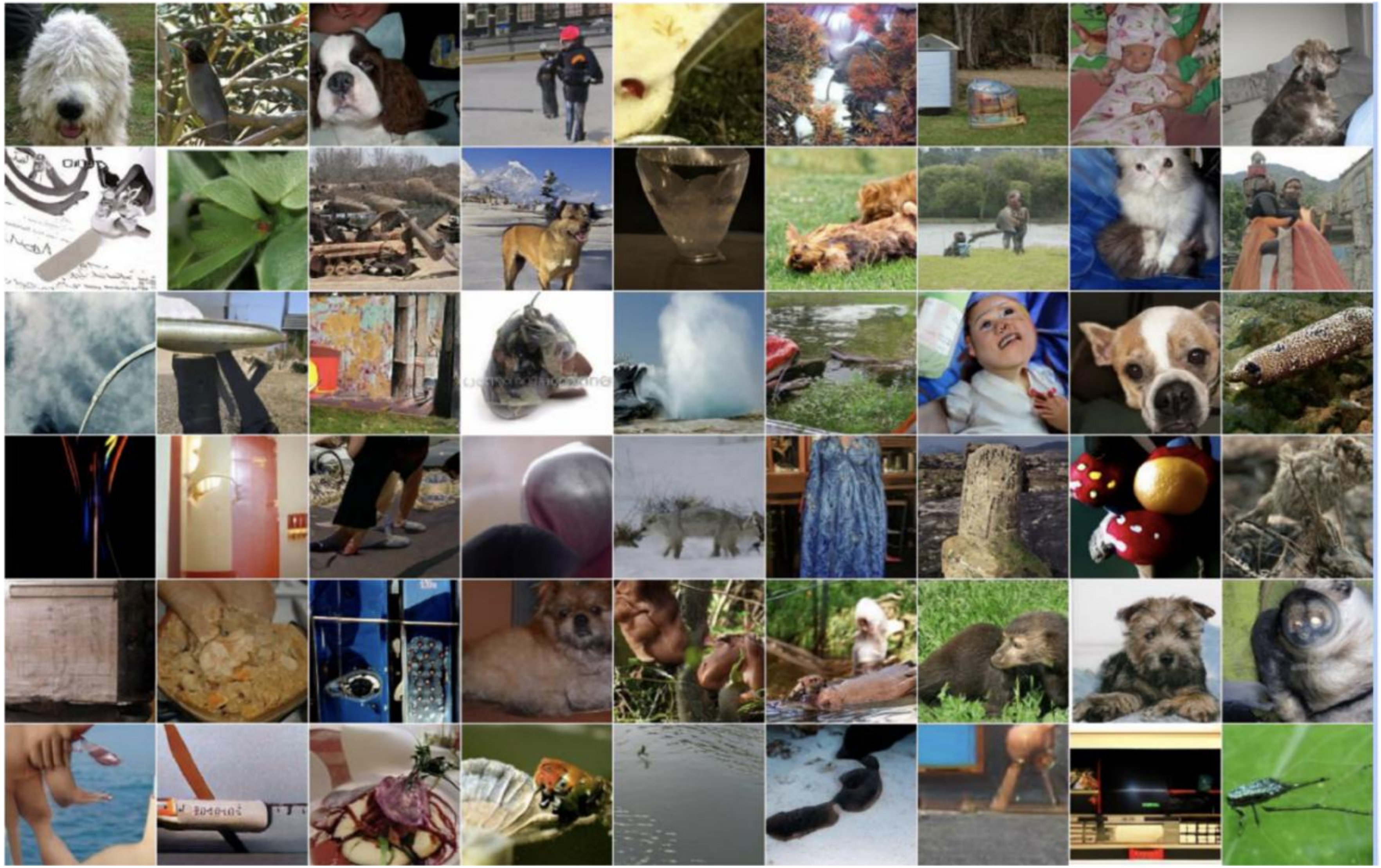
unary score terms
for x & z **only**

BigBiGAN



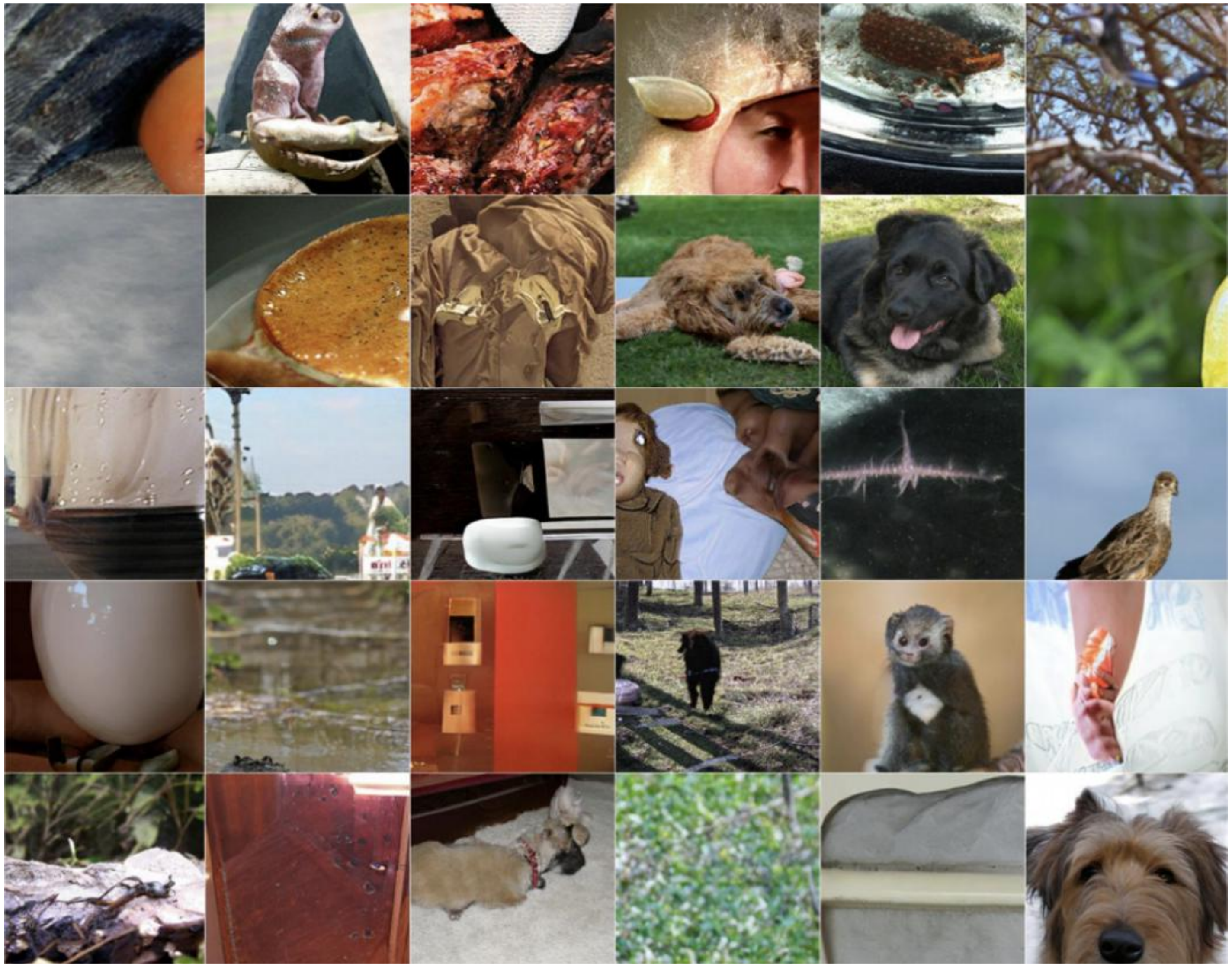
Slide credit: Jeff Donahue

BigBiGAN: Unconditional Image Generation



Slide credit: Jeff Donahue

BigBiGAN: Unconditional Image Generation



Slide credit: Jeff Donahue

BigBiGAN: Representation Learning

Method	Architecture	Feature	Top-1	Top-5
BiGAN [7, 42]	AlexNet	Conv3	31.0	-
SS-GAN [4]	ResNet-19	Block6	38.3	-
Motion Segmentation (MS) [30, 6]	ResNet-101	AvePool	27.6	48.3
Exemplar (Ex) [8, 6]	ResNet-101	AvePool	31.5	53.1
Relative Position (RP) [5, 6]	ResNet-101	AvePool	36.2	59.2
Colorization (Col) [41, 6]	ResNet-101	AvePool	39.6	62.5
Combination of MS+Ex+RP+Col [6]	ResNet-101	AvePool	-	69.3
CPC [39]	ResNet-101	AvePool	48.7	73.6
Rotation [11, 24]	RevNet-50 ×4	AvePool	55.4	-
Efficient CPC [17]	ResNet-170	AvePool	61.0	83.0
BigBiGAN (ours)	ResNet-50	AvePool	55.4	77.4
	ResNet-50	BN+CReLU	56.6	78.6
	RevNet-50 ×4	AvePool	60.8	81.4
	RevNet-50 ×4	BN+CReLU	61.3	81.9

Slide credit: Jeff Donahue

BigBiGAN: Latent Space NNs

Slide credit: Jeff Donahue

BigBiGAN Reconstructions

Computing a reconstruction $\mathbf{x}' = G(E(\mathbf{x}))$:
(1) Sample a real image $\mathbf{x} \sim P_x$
(2) Encoder predicts latents $\mathbf{z}' = E(\mathbf{x})$
(3) Generator predicts reconstruction $\mathbf{x}' = G(\mathbf{z}')$

(Big)BiGAN is not directly trained for reconstruction! =
Arises out of the objective: approx. reconstruction $\mathbf{x}' \cong G(E(\mathbf{x}))$
Optimally confuses the joint data-latent discriminator.

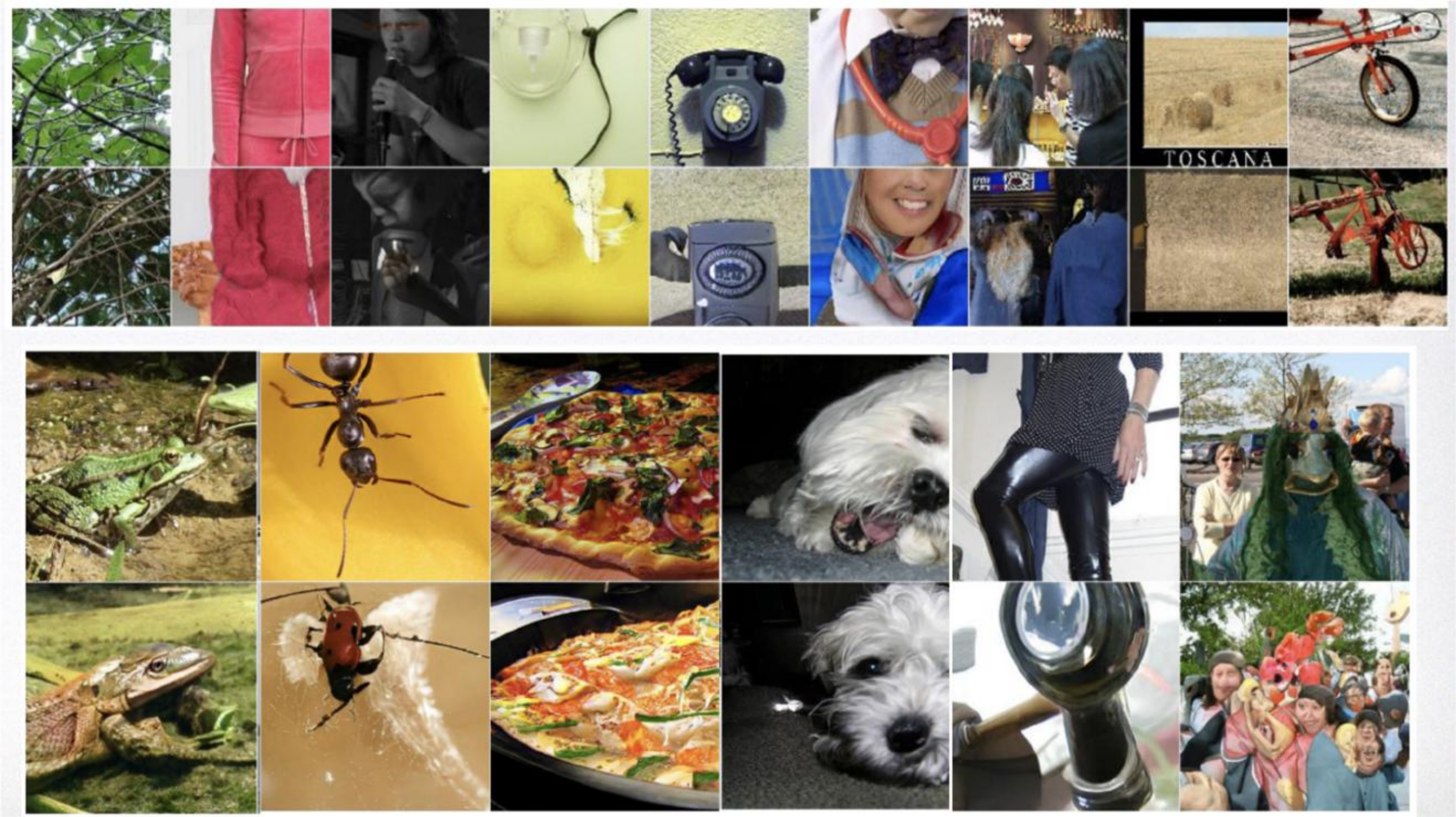
Reconstructions give insight into the semantics modeled.

real images $\mathbf{x}$

reconstructions $\mathbf{x}' = G(E(\mathbf{x}))$

Slide credit: Jeff Donahue

BigBiGAN Reconstructions



Slide credit: Jeff Donahue

Lecture overview

- Motivation and Definition of Implicit Models
- Original GAN (Goodfellow et al, 2014)
- Evaluation: Parzen, Inception, Frechet
- Theory of GANs
- GAN Progression
- Conditional GANs, Cycle-Consistent Adversarial Networks
- GANs and Representations
- **Applications**

Semi-supervised Classification (Salimans et al., 2016; Dumoulin et al., 2016)

SVNH

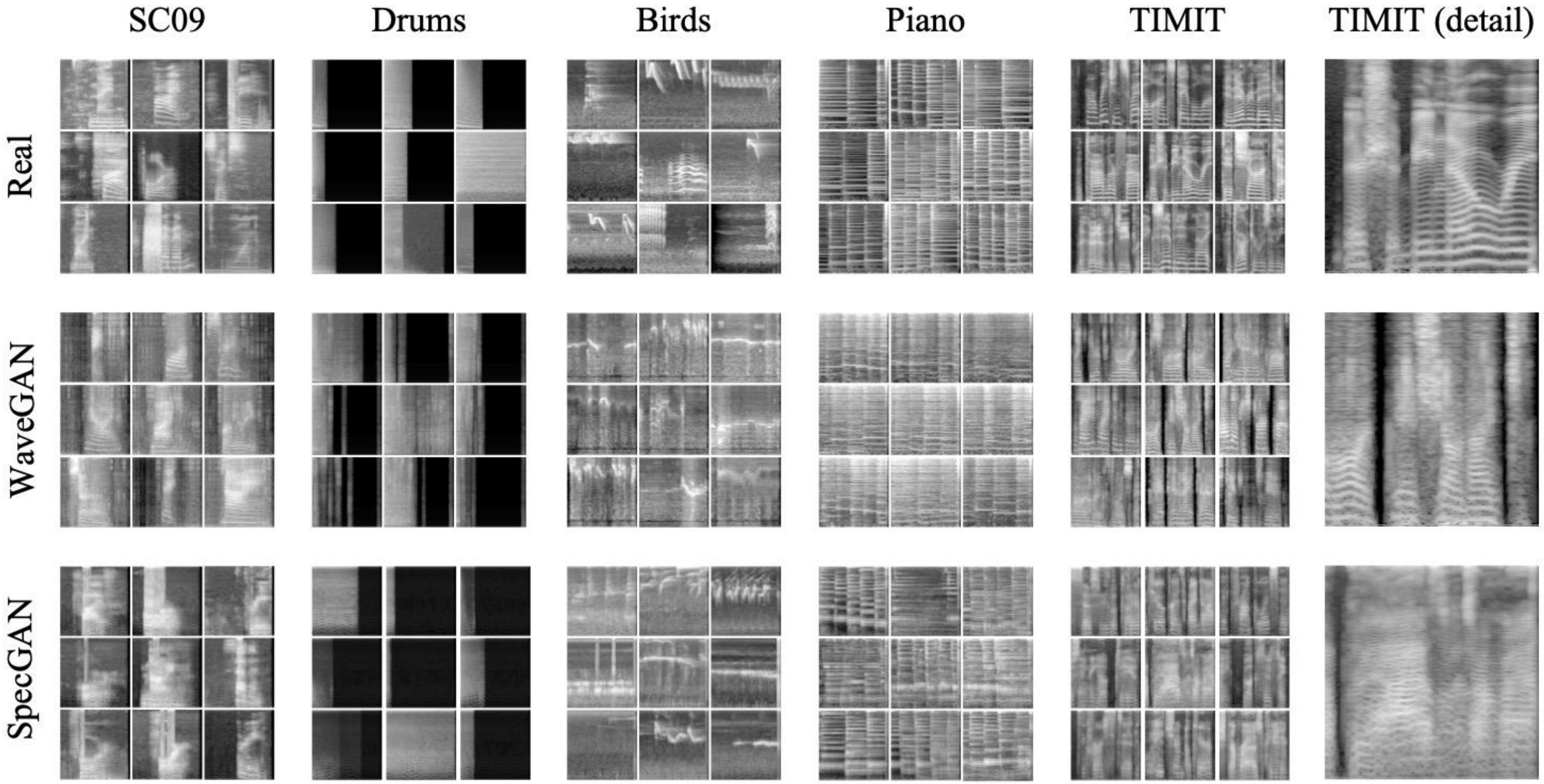
Model	Misclassification rate
VAE (M1 + M2) (Kingma et al., 2014)	36.02
SWWAE with dropout (Zhao et al., 2015)	23.56
DCGAN + L2-SVM (Radford et al., 2015)	22.18
SDGM (Maaløe et al., 2016)	16.61
GAN (feature matching) (Salimans et al., 2016)	$\mathbf{8.11 \pm 1.3}$
ALI (ours, L2-SVM)	19.14 ± 0.50
ALI (ours, no feature matching)	$\mathbf{7.42 \pm 0.65}$

Text Generation: MaskGAN (Fedus et al. 2018)

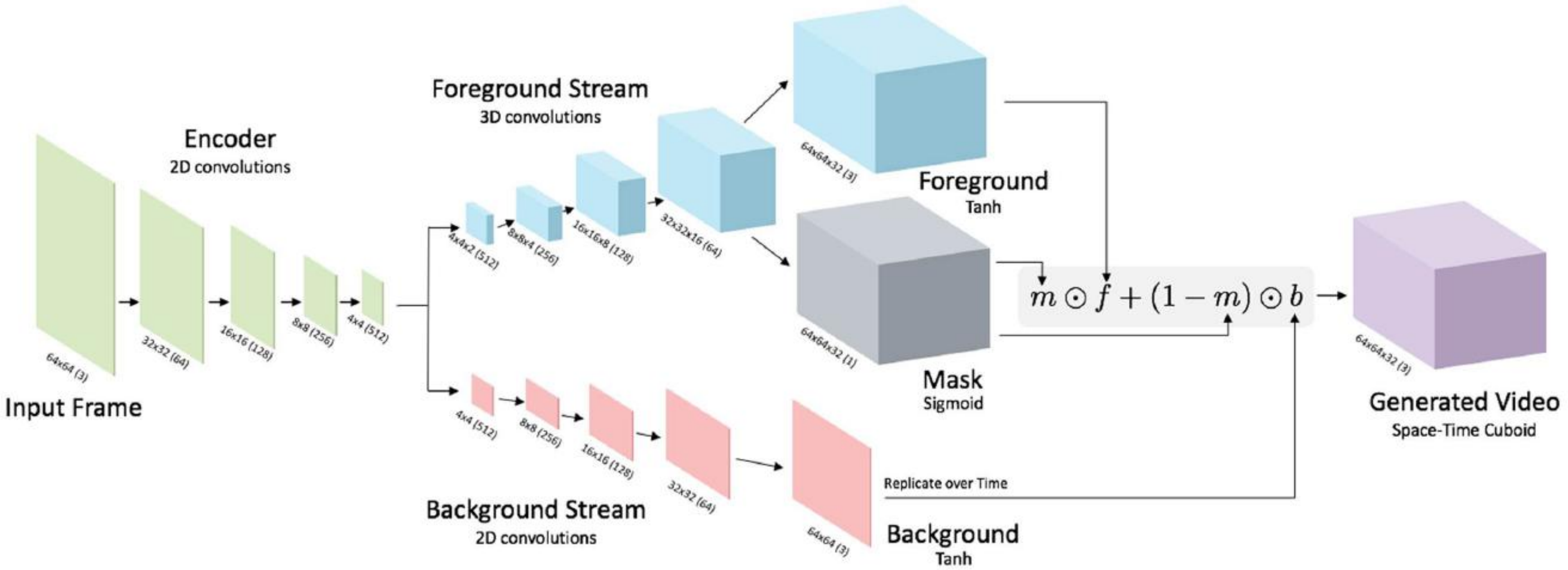
Ground Truth	**Pitch Black was a complete shock to me when I first saw it back in 2000 In the previous years I**
MaskGAN	Pitch Black was a complete shock to me when I first saw it back in <u>1979 I was really looking forward</u>
MaskMLE	Black was a complete shock to me when I first saw it back in <u>1969 I live in New Zealand</u>

Table 3: Conditional samples from IMDB for both MaskGAN and MaskMLE models.

Audio Synthesis: WaveGAN (Donahue et al. 2020)

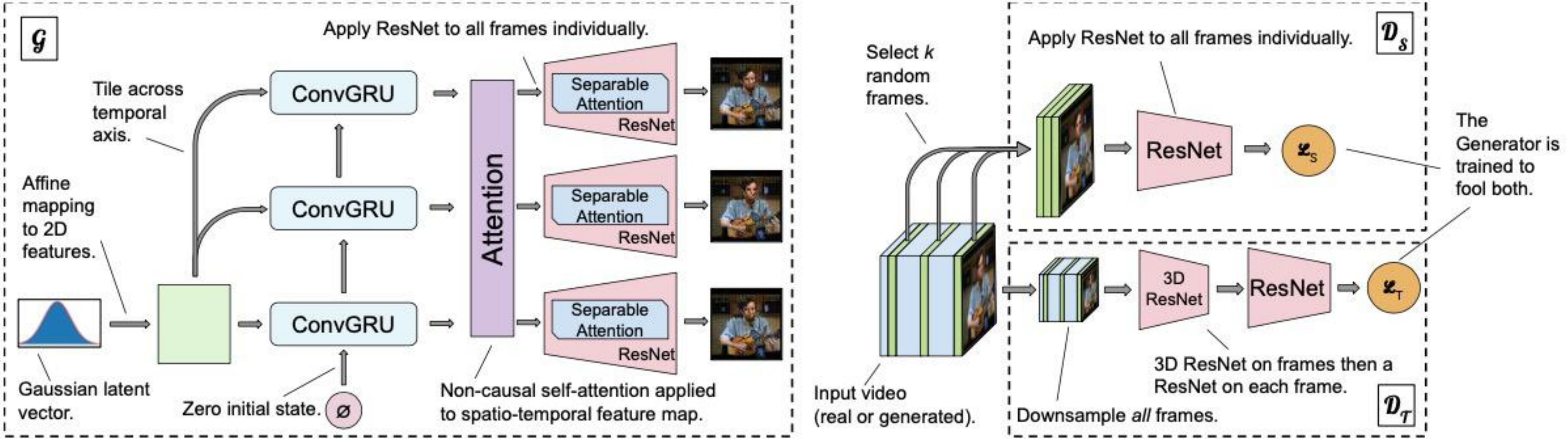


Video Generation (Vondrick et al., 2016)

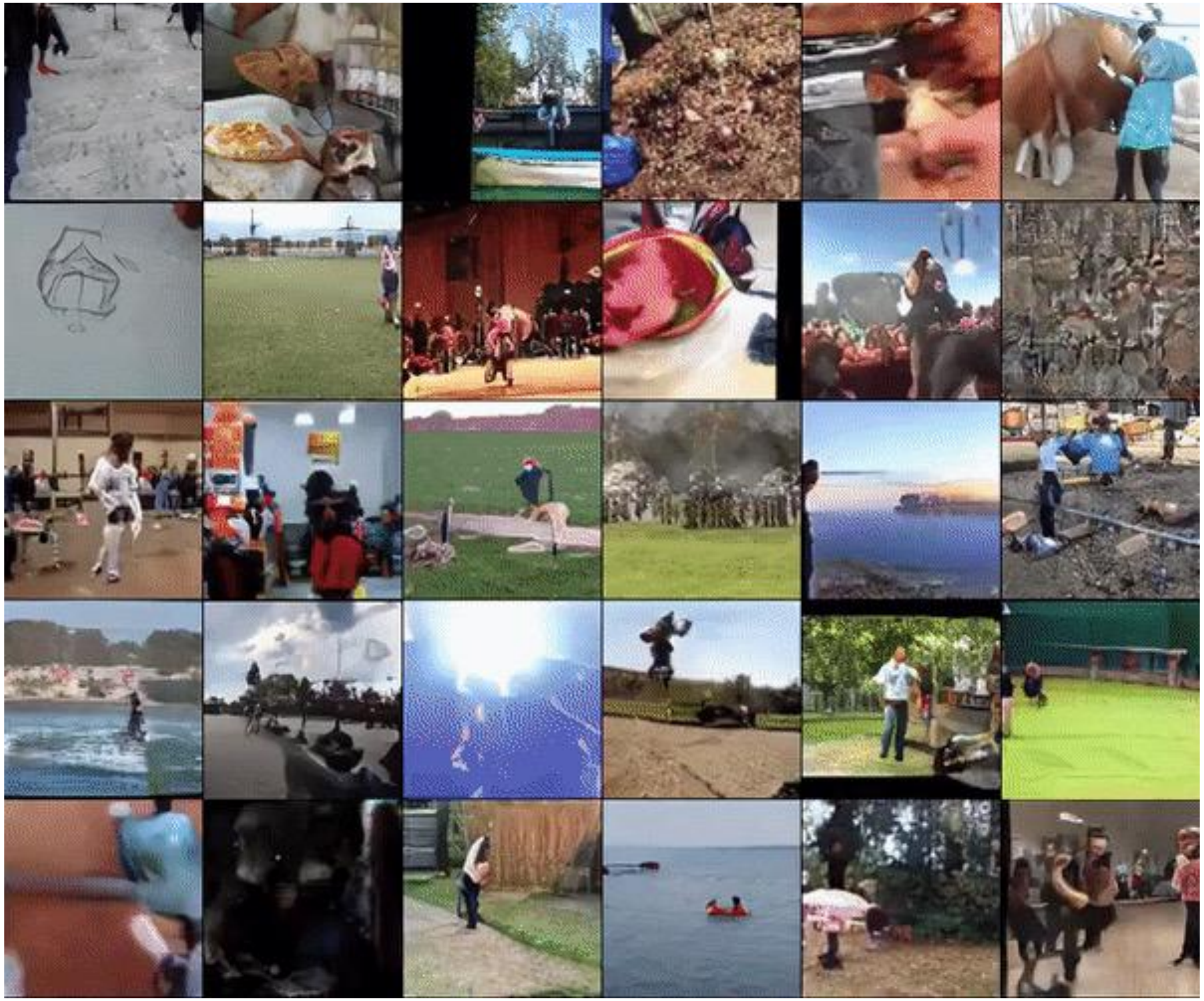


Beach Golf Train Station

DVD-GAN: Efficient Video Generation
(Clark et al., 2019)

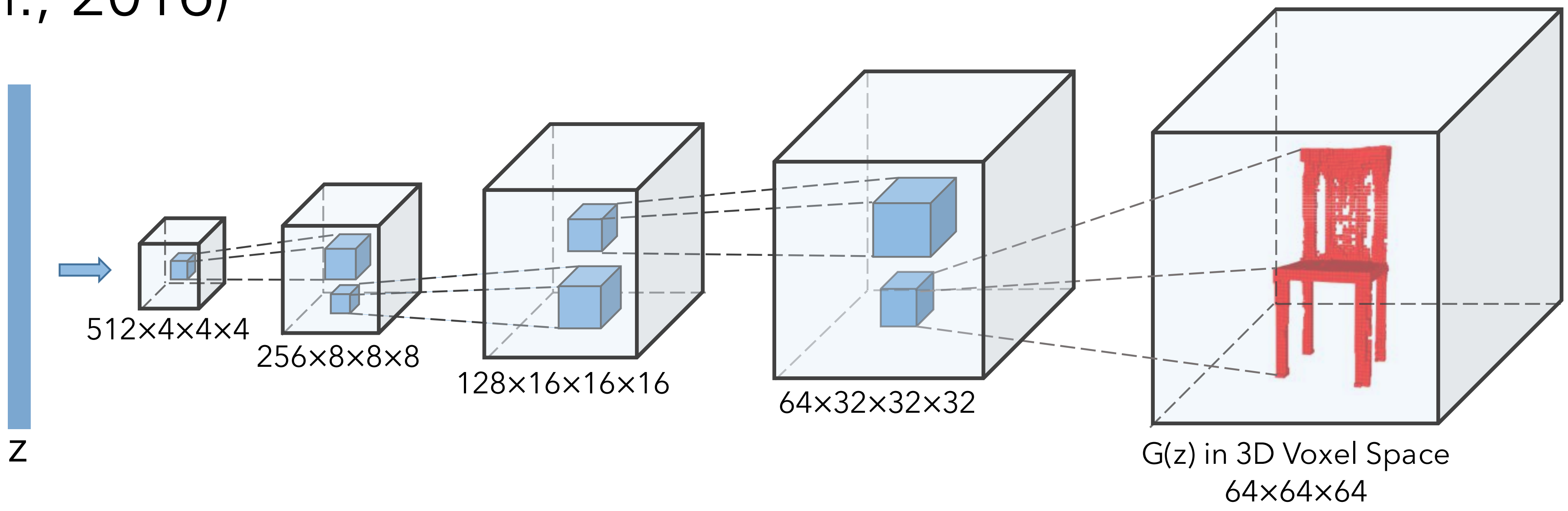


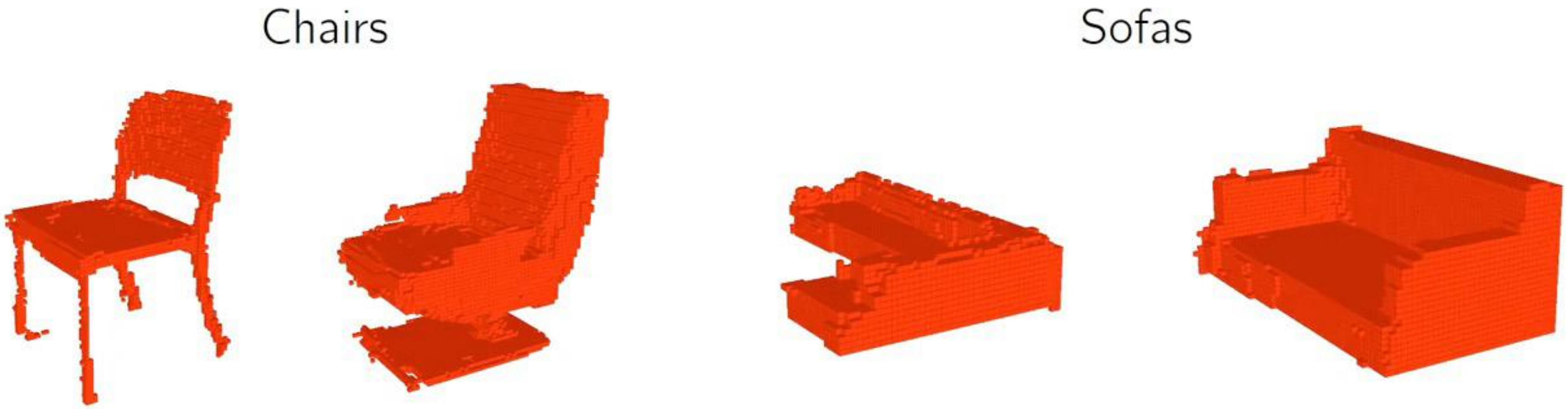
DVD-GAN: Efficient Video Generation
(Clark et al., 2019)



3DGAN: Generative Shape Modeling

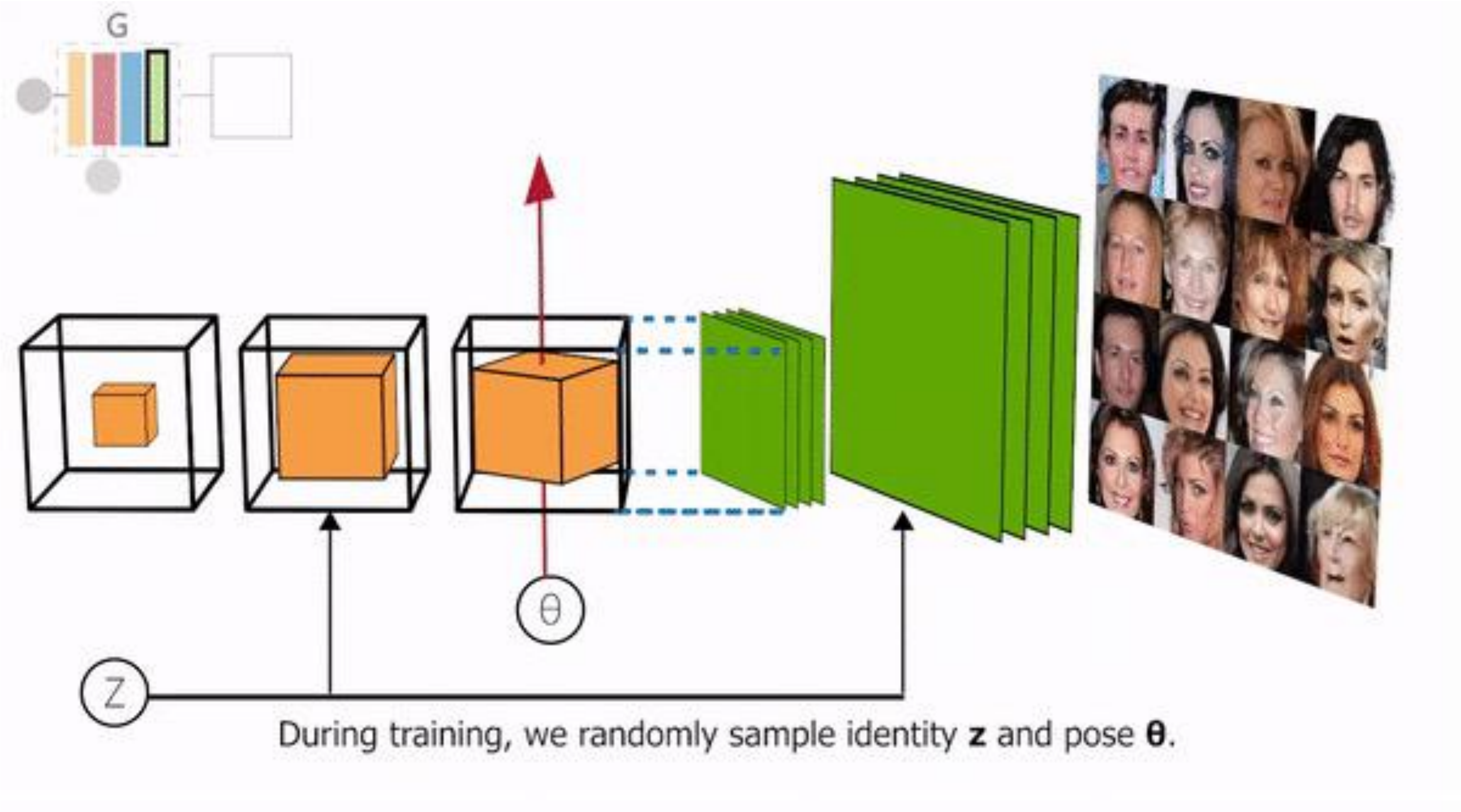
(Wu et al., 2016)





HoloGAN: Learning 3D Representations from Images

(Nguyen-Phuoc et al., 2020)



HoloGAN: Learning 3D Representations from Images

(Nguyen-Phuoc et al., 2020)

Motion Transfer: Everybody Dance Now

Vid2Vid: Video to Video Synthesis

StackGAN: Text-to-Image Synthesis (Zhang et al.'16)

The small bird has a red head with feathers that fade from red to gray from head to tail

The petals of this flower are white with a large stigma

A unique yellow flower with no visible pistils protruding from the center

This flower is pink and yellow in color, with petals that are oddly shaped

This is a light colored flower with many different petals on a green stem

This flower is yellow and green in color, with petals that are ruffled

The flower have large petals that are pink with yellow on some of the petals

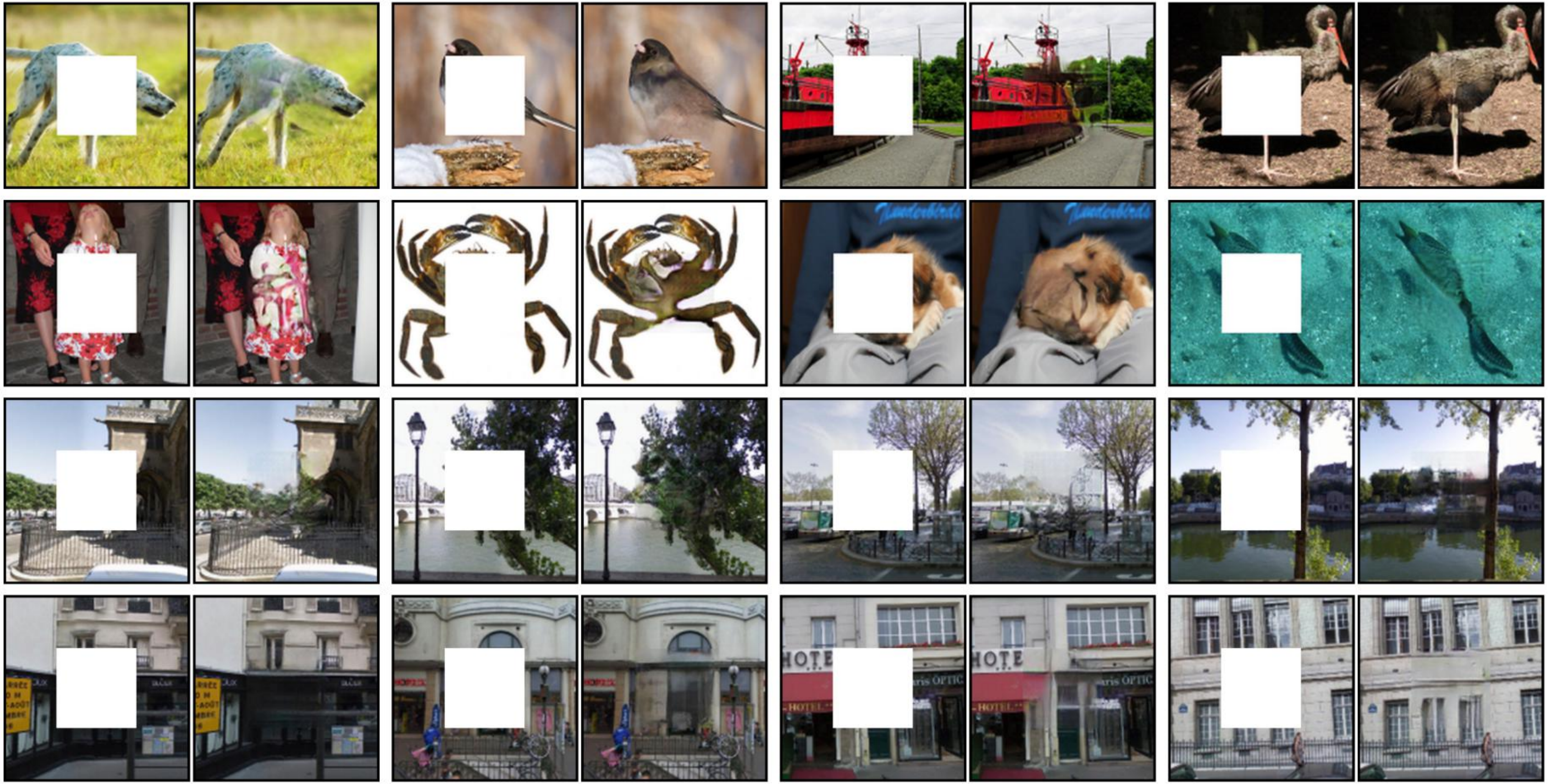
A flower that has white petals with some tones of yellow and green filaments

SRGAN: Single Image Super-Resolution

(Ledig et al., 2017)

- Combine content loss with adversarial loss

Image Inpainting (Pathak et al., 2016)



Unsupervised Domain Adaptation (Bousmalis et al., 2016)

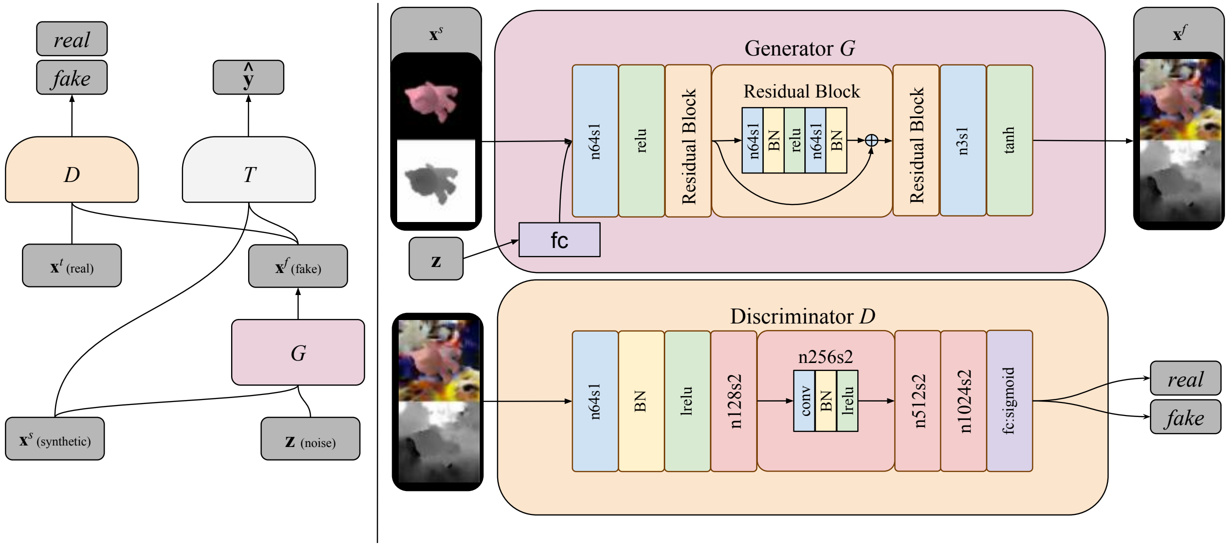
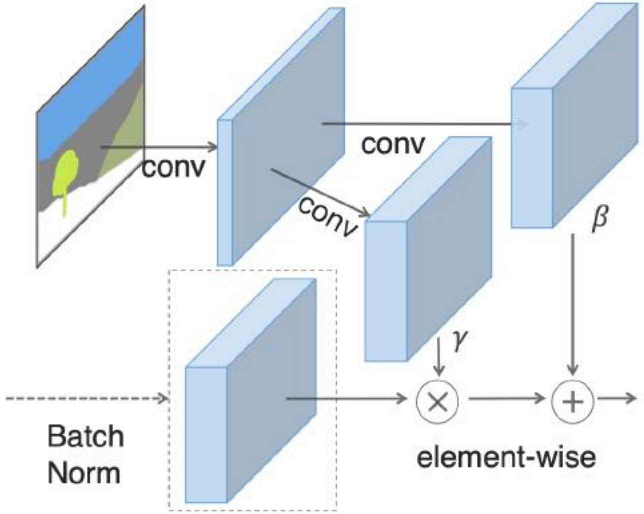


Image examples from the Linemod dataset

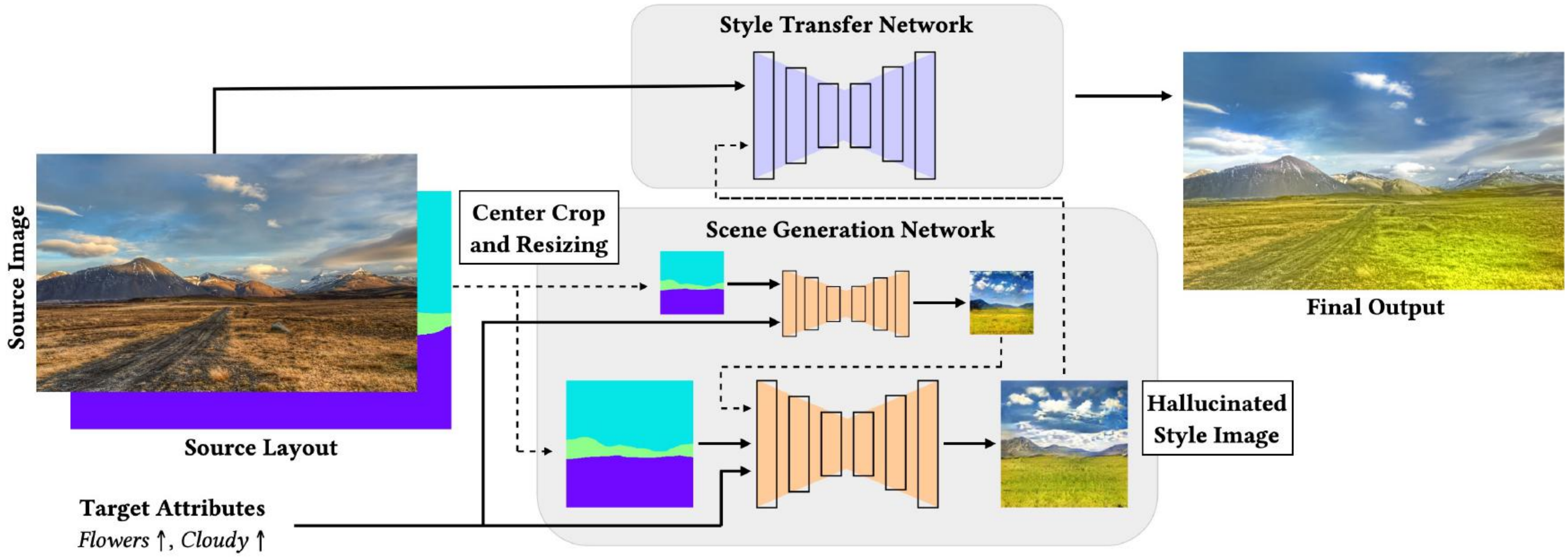
RGDB image samples
(conditioned on a synthetic image)

Semantic Image Editing: GauGAN



(Park et al. 2019)

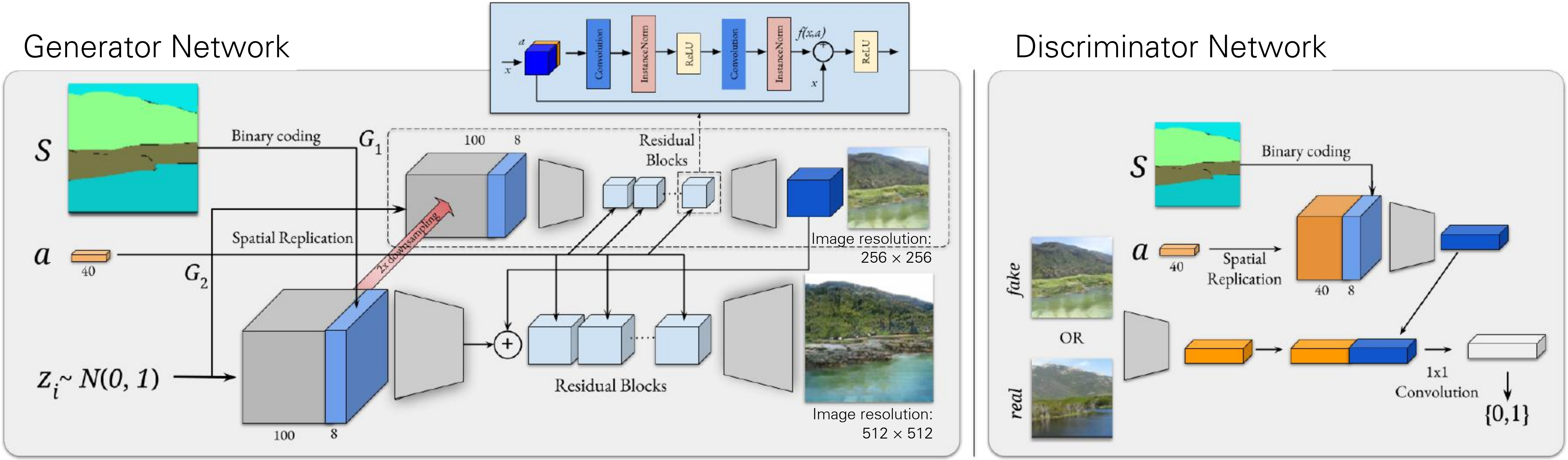
Semantic Image Editing (Karacan et al. 2020)



https://hucvl.github.io/attribute_hallucination/

Scene Generation Network (SGN)

- The semantic layout categories are encoded into 8-bit binary codes
- The transient attributes are represented by a 40-d vector.



- An architecture similar to Pix2pixHD model (Wang et al. 2018)
- **Generator network**: A coarse-to-fine model with 2 generator networks
- **Discriminator network**: A combination of three different discriminator networks operating at an image pyramid of 3 scales

T.-C. Wang et al. **High-resolution image synthesis and semantic manipulation with conditional GANs.** CVPR 2018.

Training Objective of SGNs

$$\mathcal{L}_{SGN} = \min_{G}\left(\left(\max_{D=\{D_1,D_2,D_3\}} \sum_{k=1,2,3} \mathcal{L}_{GAN}\left(G, D_k\right)\right) + \lambda \mathcal{L}_{percep}(G)\right)$$

- Relative Negative Mining (RNM)
  - real image, relevant attributes and layout

    vs.

    fake image, relevant attributes and layout
    real image, mismatching layout (chosen from hard negatives)
    or mismatching attributes

- Layout-Invariant Perceptual Loss
  - $\mathcal{L}_{percep}(G) = E_{z\sim p_z(z);x,S,a\sim p_{data}(S,a)}\left[\|f_P(x) - f_P(G(z,a,S))\|_2^2\right]$
  - f_P: CNN encoder for the scene parser network (Zhou et al., 2018)

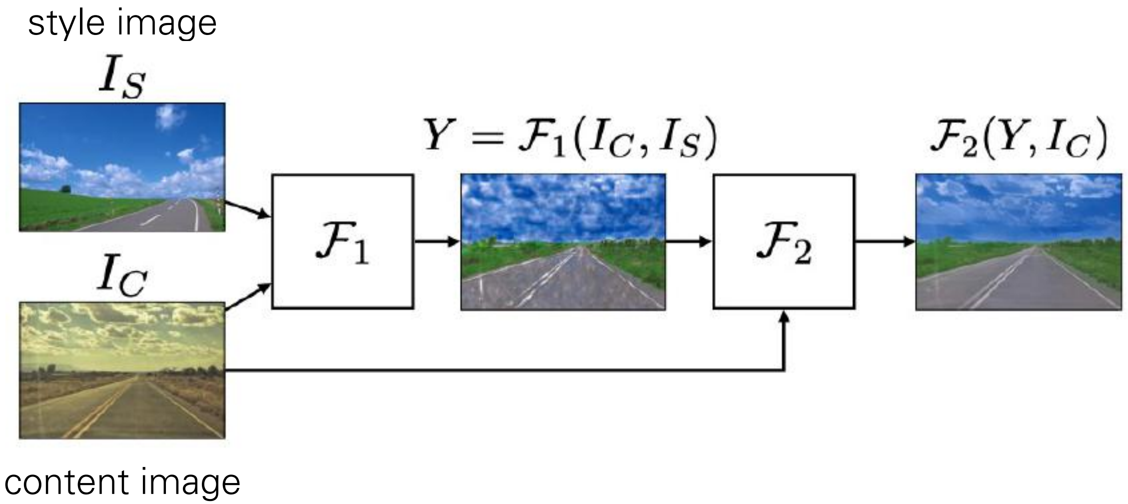
B. Zhou, H. Zhao, X. Puig, S. Fidler, A. Barriuso, A. Torralba. **Scene Parsing through ADE20K Dataset.** CVPR 2017.

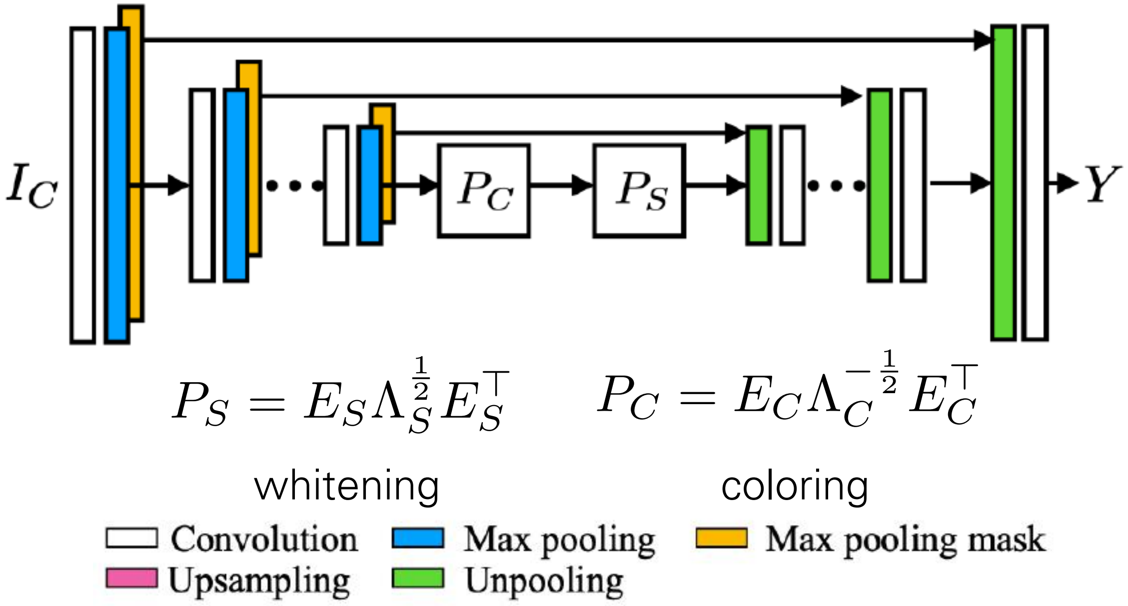
Style Transfer Network

- The FPST method of (Li et al., 2018), which is composed of two steps with close-form solutions:
  1. Stylization step $\mathcal{F}_1$
  2. Smoothing step $\mathcal{F}_2$

  $$I_{out} = \mathcal{F}_2\left(\mathcal{F}_1(I_C, I_S), I_C\right)$$

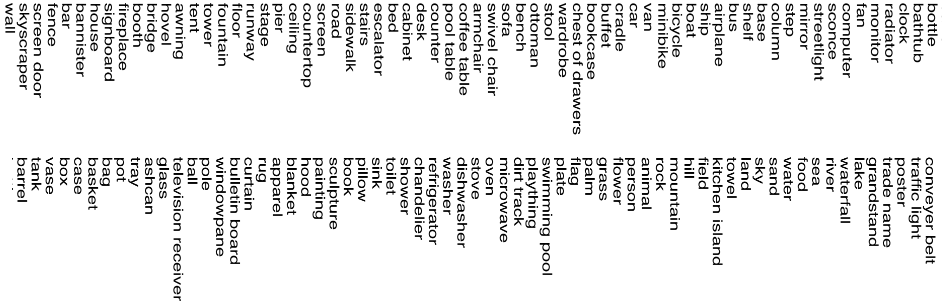
- The **stylization step** is based on the whitening and coloring transform to stylize images via feature projections
  - Style information encoded by the covariance matrix of VGG features
- The **smoothing step** ensures spatially consistent stylizations via a manifold ranking operator.



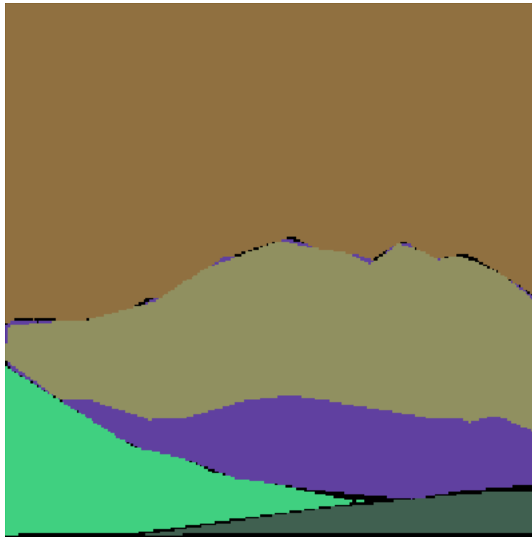


ALS18K Dataset

- A dataset of **17772 outdoor images** with layout and transient attribute labels, formed by combining and annotated images from
  - Transient Attributes dataset (Laffont et al., 2013)
  - ADE20K dataset (Zhou et al., 2017)
- 16434 images for training, 1338 images for testing
- **150 semantic categories**
- **40 transient attributes** in five categories



lighting: sunrise/sunset, bright, daylight, etc.
weather: sunny, warm, moist, foggy, cloudy, etc.
seasons: spring, summer, autumn, winter
subjective impressions: gloomy, soothing, beautiful, etc.
additional attributes: active/busy, cluttered, dirty/polluted, lush vegetation, etc.



P.-Y. Laffont, Z. Ren, X. Tao, C. Qian, J. Hays. **Data-driven Hallucination of Different Times of Day from a Single Outdoor Photo**. SIGGRAPH Asia 2013.
B. Zhou, H. Zhao, X. Puig, S. Fidler, A. Barriuso, A. Torralba. **Scene Parsing through ADE20K Dataset.** CVPR 2017.

Semantic layout

Manipulating Attributes of Natural Scenes via Hallucination [Karacan et al., 2020]

Manipulating Attributes of Natural Scenes via Hallucination [Karacan et al., 2020]

Manipulating Attributes of Natural Scenes via Hallucination [Karacan et al., 2020]

Manipulating Attributes of Natural Scenes via Hallucination [Karacan et al., 2020]

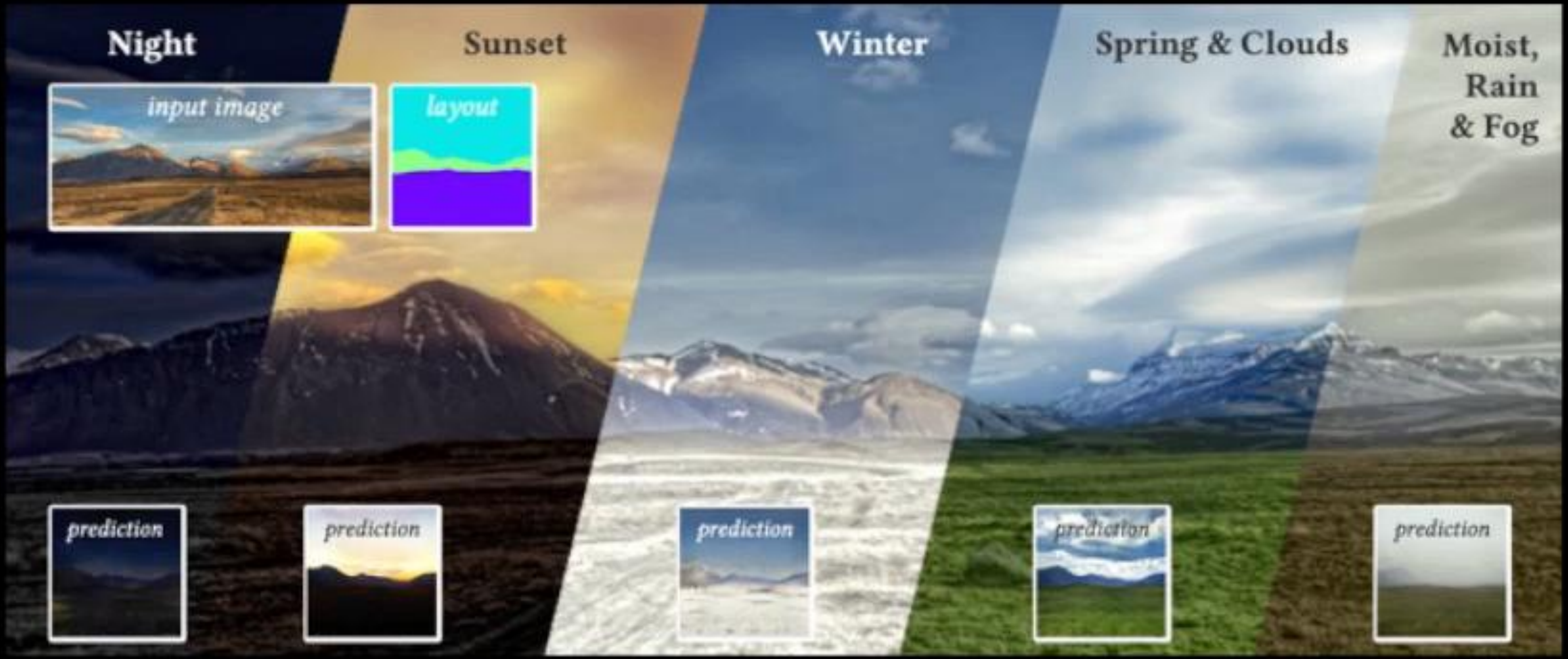
Manipulating Attributes of Natural Scenes via Hallucination [Karacan et al., 2020]

Manipulating Attributes of Natural Scenes via Hallucination [Karacan et al., 2020]

Manipulating Attributes of Natural Scenes via Hallucination [Karacan et al., 2020]

Manipulating Attributes of Natural Scenes via Hallucination [Karacan et al., 2020]

Manipulating Attributes of Natural Scenes via Hallucination [Karacan et al., 2018]

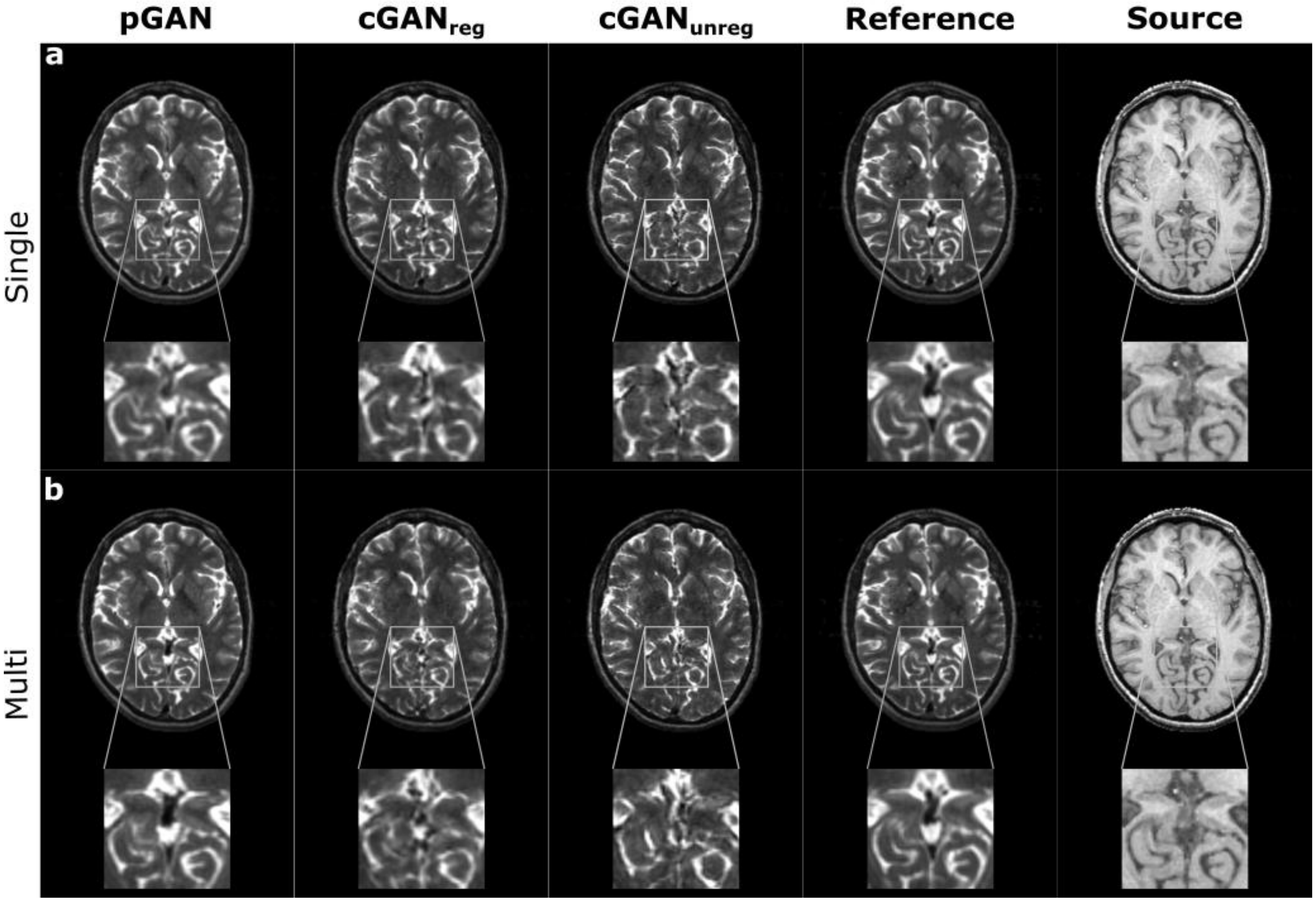


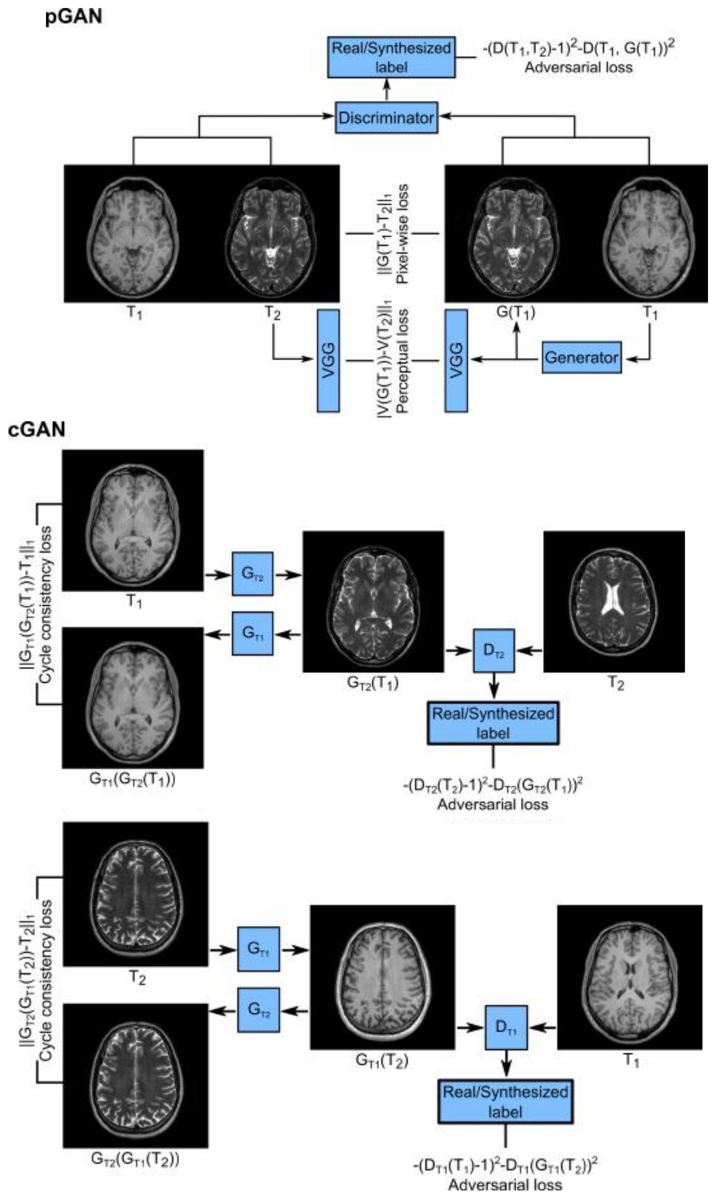
Manipulating Attributes of Natural Scenes via Hallucination

Levent Karacan, Zeynep Akata, Aykut Erdem, Erkut Erdem

ACM Transactions on Graphics

Demo





- Image Synthesis in Multi-Contrast MRI [Ul Hassan Dar et al. 2019]

Next lecture:
Score-Based and
Denoising Diffusion Models